SEPEC Conference
PROCEEDINGS

hypermedia & information reconstruction

Aerospace Applications & Research Directions

December 3 - 5, 1990

Houston, Texas

South Shore Harbour Resort & Conference Center

Co-Sponsored by

University of Houston-Clear Lake
NASA/Johnson Space Center
Hypermedia Working Group

hypermedia & information reconstruction

Aerospace Applications & Research Directions

SEPEC Conference Proceedings

December 4 & 5, 1990

Houston, Texas

South Shore Harbour Resort & Conference Center

Co-Sponsored by

University of Houston-Clear Lake

NASA/Johnson Space Center

Hypermedia Working Group

Hypermedia '90

Hypermedia and Information Reconstruction

Hypermedia '90 is yet another effort to highlight the power and usefulness of a rapidly emerging technology. The theme of the conference, "Hypermedia and Information Reconstruction," was chosen to underscore the influence of hypermedia on information technology. The awareness of a multidimensional information space, open to linking and browsing, has transformed the way we store. retrieve and use information. The sessions, presented by distinguished professionals from government, industry and university backgrounds, explore the subtleties and impact of this transformation.

The sessions included in the proceedings are representative of a wide variety of hypermeida uses and applications . The sessions focus, for the most part, on real-world hypermeida projects, represented largely by aerospace applications, and point to some future directions in hypermedia research and development. The concepts presented here are bold and innovative. They do indeed represent the growing power and utility of an exciting new technology.

On behalf of the Hypermedia Working Group, welcome to the conference.

Charles Hardwick
Executive Chair

Hypermedia '90

Aerospace Applications & Research Directions

In 1988, the first Hypermedia Conference sponsored by NASA/JSC and UH-Clear Lake focused on a new technology emerging within the aerospace community. Only two years later we have witnessed remarkable changes in hypermedia technology. A growing number of hardware and software tools have made the linking of information from various sources much easier. Applications have grown richer in content and bolder in concept.

Hypermedia allows nonlinear access to information by following links between nodes. New tools allow developers to link 'chunks' of information in a wide variety of forms -- text, code, animations, images, sound, full-motion video. The prospect of creating a multidimensional information space designed to enhance communication and understanding is rapidly becoming a reality.

The theme of our second Hypermedia Conference is: "Hypermedia and Information Reconstruction." *The concept of electronic documentation, for example, has taken on new dimensions with the introduction of multimedia. Intelligent computer based training utilizing hypertext, expert systems, and multimedia devices offers an expanded range of exciting possibilities. The conference will provide both a look into these and other new areas, and an opportunity to interact with colleagues sharing common interest in hypermedia tools and research.*

The tutorials and conference look at the "State of the practice," with a focus on aerospace applications and some future directions in hypermedia research and development. Sessions have been built around presentations of working systems, systems currently under development, and research activities. Application areas include training, electronic documentation, design knowledge capture, configuration and project management, and interface applications.

Exhibits & Demonstrations

Exhibits and demonstrations are being assembled to showcase leading edge hypermedia and multimedia products and applications such as text, code, animations, images, sound, and full-motion video. Vendors and end-users will present the enabling capabilities of these emerging technologies.

The tools and products will be shown throughout the conference. Demonstrations are being chaired by Ed Kusik, (713) 282-3506. Exhibits are being chaired by Kathy Kirchner, (713) 283-2950.

Hypermedia Steering Committee

Executive Chair
Charles S. Hardwick, *University of Houston-Clear Lake*
Technical Co-Chairs
Dona Erb, *MITRE*
Barbara Kolkhorst, *IBM*
Tutorial Chair
Chris Dede, *University of Houston-Clear Lake*
Administrative Chair
Glenn B. Freedman, *University of Houston-Clear Lake*
Demonstrations Chair
Ed Kusik, *Rockwell Space Operations Company*
Exhibits Chair
Kathy Kirchner, *University of Houston-Clear Lake*
Steering Committee Members
David Proctor, *NASA/Johnson Space Center*
Ken Crouse, *NASA/Johnson Space Center*
Frank Hughes, *NASA/Johnson Space Center*
Linda Christman, *NASA/Johnson Space Center*
Bob Savely, *NASA/Johnson Space Center*
Glen Van Zandt, *NASA/Johnson Space Center*
Robert MacDonald, *NASA/Johnson Space Center*
Eric Lloyd, *University of Houston-Clear Lake*
Administrative Staff
Don Myers, *SEPEC, University of Houston-Clear Lake*
Bette Benson, *SEPEC, University of Houston-Clear Lake*
Mary Watson, *SEPEC, University of Houston-Clear Lake*
Pat Williams, *SEPEC, University of Houston-Clear Lake*
Patricia Vining, *SEPEC, University of Houston-Clear Lake*

CALENDAR

Monday, December 3

8:00 a.m.	Tutorial Registration
8:30 -Noon	Tutorials
Noon	Lunch
1:30 p.m.- 5:00	Tutorials

Tuesday, December 4

8:00 a.m.	Registration
8:30	Welcome & Introductions
8:45	Keynote Speaker
10:00-5:00 p.m.	Exhibits & Demonstrations
10:15	Session 1
Noon	Lunch
1:30 p.m.	Session 2
3:15	Session 3
5:00	Wine & Cheese Reception

Wednesday, December 5

8:00 a.m.	Registration
8:30	Session 4
10:00-5:00 p.m.	Exhibits & Demonstrations
10:15	Session 5
Noon	Lunch
1:30 p.m.	Session 6
3:15	Session 7
5:00	Closing

December 4, Tuesday

8:30 - 8:45 a.m. **WELCOME & INTRODUCTIONS**

Charles S. Hardwick
Conference Executive Chair & Professor
University of Houston-Clear Lake

Thomas M. Stauffer
President
University of Houston-Clear Lake

Robert T. Savely
Chief, Software Technology Branch
NASA/ Johnson Space Center

Glenn B. Freedman
Associate Vice President, Academic Initatives
University of Houston-Clear Lake

Keynote Address

8:45 - 10:00 a.m.

Introduced by **Barbara Kolkhorst**, IBM Corporation

Patricia Ann Carlson

Center For Advanced Media, International Centers For Telecommunication Technology

Rose-Hulman Institute of Technology

Hypertext, Typographic Man, and the Notion of Literacy

Patricia Carlson is currently a National Research Council Senior Associate at the Human Resources Laboratory, Intelligent Systems Branch, Brooks AFB, San Antonio, Texas, researching the integration of intelligent hypertext systems and neural networks. Her current areas of interest include hypertext as a form of knowledge representation and AI applications in the areas of knowledge management systems, intelligent tutoring systems, and virtual world technology.

Break

10:00 - 10:15 a.m.

Session 1

10:15 - Noon **STATE OF THE PRACTICE**

Chair: **David Proctor,** *Life Sciences, NASA/Johnson Space Center*

Evolution of the Experiment Document Information System

Jane Moorhead
Henry Brans
Houston Applied Logic & Life Sciences Project Division, NASA/Johnson Space Center

A history of the evolution of LSPD Experiment Document Automation, including strengths and weaknesses of the old Vax-based versus new HyperCard-based systems, plus why HyperCard was selected and plans for the future.

Engineering Software Development with HyperCard

Robert J. Darko
Hudson Products Corporation

Description of successful and unsuccessful techniques used in the development of stackware for use in an engineering environment. Plus an evaluation of the use of HyperCard in engineering applications.

Life Sciences On-Line: A Study in Hypermedia Application

Linda Christman
Nam Hoang
Life Sciences, Project Division, NASA/Johnson Space Center

A demonstration of an on-line training prototype module using the in-flight flow cytometer, an instrument which measures cells. The HyperCard stack illustrates the use of hypermedia to recall information required to perform Life Sciences experiments at -0- gravity. Results of a survey of hypermedia developers and lessons learned during development of the module will be presented.

Lunch

Noon - 1:30 p.m.

Session 2

1:30 - 3:00 p.m. **EDUCATION AND TRAINING: DIRECTIONS IN HYPERMEDIA**

Co-Chair: **Glenn B. Freedman,** *University of Houston-Clear Lake*

Co-Chair: **Glen Van Zandt,** *Human Resources Development, NASA/Johnson Space Center*

Computer-Assisted Knowledge Acquisition for Hypermedia Systems

Kurt Steuck

Air Force Human Resources Laboratory, Brooks AFB

A description of how procedural and declarative knowledge can be used to set up the structure or "web" of a hypermedia environment.

A Knowledge Base Browser Using Hypermedia

Tony Pocklington
McDonnell Douglas Space Systems

Lui Wang
NASA/Johnson Space Center

Discussion of a hypermedia system we are developing to browse CLIPS knowledge bases. This system will be used to help train flight controllers for the Mission Control Center.

A Model for Addressing Navigation Limitations and Metacognitive Constraints in Hypermedia Training Systems

Glenn B. Freedman

University of Houston-Clear Lake

Hypermedia training systems are often limited by the practical limitations of the navigation systems of the tools and the metacognitive constraints of the users and developers. This presentation describes a model for hypermedia systems developers to consider in building new systems or maintaining existing systems.

Break

3:00 - 3:15 p.m.

Session 3

3:15 - 5:00 p.m. **ISSUES FOR REAL-WORLD HYPERTEXT PROJECTS**

Chair: **Robert J. Glushko,** *Search Technology, Inc.*

Panel: Three Issues for Real-World Hypertext Projects

Robert J. Glushko
Search Technology, Norcross, GA

David Gunning
Human Resources Laboratory, Wright-Patterson AFB, OH

Bruce Warren
Warren-Forthought, Angleton, TX

Wine & Cheese Reception

5:00 - 6:30 p.m.

Wednesday, December 4
Session 4
8:30 a.m. - 10:00 a.m. **RESEARCH ACTIVITIES AT NASA**

Chair: **Robert T. Savely,** *NASA/Johnson Space Center*

Hypermedia Applications in a Mission Operations Environment

Troy J. Ames
NASA/Goddard Space Flight Center

The Automation Technology Section at Goddard Space Flight Center is investigating hypermedia applications to support all phases of mission operations. This presentation will highlight a few specific hypermedia application areas such as computer aided instruction of flight operations teams, control center software development tools, computer-human interface design tools, and control center operations.

Integrating Knowledge and Control into Hypermedia-Based Training Environments: Experiments with HyperCLIPS

Randall W. Hill, Jr.
Jet Propulsion Laboratory

Discussion of issues in knowledge representation and control in Hypermedia–based training environments; a knowledge representation structure called a "concept network"; and finally the tool we have developed called HyperCLIPS.

GERM as a Tool for Space Station Documentation

Ken Crouse
NASA/ Johnson Space Center

Charles S. Hardwick
University of Houston-Clear Lake

The presentation will discuss the results of an investigation to examine the use of a hypermedia system for representing the interrelationships among space station documentation objects and providing a flexible graphic interface for viewing and navigating through the document hierarchy.

Break
10:00 - 10:15 a.m.

Session 5
10:15 a.m. - Noon **INTERFACES FOR HYPERMEDIA SYSTEMS**

Chair: **Dona Erb,** *MITRE Corporation*

Hypertext As a Model for the Representation of Computer Languages

Randal Davis
University of Colorado

Computer systems for operating the Space Station Freedom will include an object-oriented and English-like User Interface Language (UIL). We have proposed a representation of the Space Station UIL that is based on a hypertext model. We discuss the hypertext model of the Space Station UIL and show how this representation may be appropriate for other modern computer languages.

Automating Hypertext in a Decision Support System

Michael Bieber
Boston College

Hypertext typically requires manually predefined connection (links). Our research, concerning decision support system (DSS) shells serving arbitrary DSS applications (scientific, business oriented, etc.) demands constructing hypertext connections "on the fly". We are automating hypertext for DSS application exploration, execution and report generation.

TEJAS: Hypermedia for the NASA Masses

Michael L. Drews
NASA Headquarters

This presentation will not be about TEJAS. It will be about creating, developing, distributing and sustaining TEJAS as a novel hypermedia application for NASA. This will involve a hypermedia presentation that details the steps, challenges, pitfalls, rewards and other consequences.

Lunch
Noon - 1:30 p.m.

Session 6
1:30 - 3:00 p.m.

HYPERTEXT AND OBJECT MANAGEMENT

Chair: **Bryan Fugate,** *Microelectronics and Computer Technology Corporation (MCC)*

AI GERM: A Logic Programming Front End for GERM

Safaa H. Hashim
Microelectronics and Computer Technology Corporation (MCC)

HEAVENS System for Software Artifacts

Paul Matthews
Bellcore Company

The HEAVENS system is a workstation-based collection of software for analyzing, organizing and viewing software artifacts. As a prototype, the system has been used for visualizing source code structure, analyzing dependencies, and restructuring to simplify maintenance. The system has also been used in the early stages of software design to organize and relate design objects, maintain design documentation, and provide a ready-made framework for later coding.

Break
3:00 - 3:15 p.m.

Session 7
3:15 - 5:00 p.m.

FUTURE OF HYPERMEDIA

Chair: **David Palumbo,** *University of Houston- Clear Lake*

Hypermedia as Medium

Chris Dede
University of Houston- Clear Lake

Since the dawn of civilization, few full-fledged media have emerged as vehicles for thought and interaction (the spoken word, the written word, still images, full-motion images). Enthusiasts for hypermedia are now claiming that a new medium has been developed to symbolize and communicate ideas. Few technological innovations have greater potential to transform society than a new medium (e.g. the long-term effects of the invention of writing). Will hypermedia be the first computer-centered medium?

Hypermedia = Hypercommunication

Mark R. Laff
IBM - T.J. Watson Research Center

New hardware and software technology have given application designers the freedom to use new realism in human computer interaction. High-quality images, motion video, stereo sound and music, speech, touch, gesture, provide richer data channels between the person and the machine. Ultimately, this will lead to richer communication between people with the computer as an intermediary. The whole point of hyperbooks, hyper-newspapers, and virtual worlds is to transfer the concepts and relationships, the "data structure" if you will, from the mind of the creator to that of the user. In this presentation we will discuss some of the characteristics of this rich information channel followed by some examples of our work, including an interactive hypermedia biography of IBM Fellow John Cocke entitled "John Cocke: A Retrospective by Friends."

Moving From Knowledge Presentation To Knowledge Representation

David B. Palumbo
University of Houston- Clear Lake

Nodes and links are the common terminology of hypermedia, cognitive psychology, and artificial intelligence. However, upon closer inspection these disciplines are not talking about the same thing. The focus of this presentation will be on moving the node-link framework of hypermedia closer to that of cognitive psychology and artificial intelligence.

Closing
5:00 p.m.

TUTORIALS

December 3, 8:30 a.m. - Noon

"Structured Hypertext"

J. Scott Johnson

NTERGAID

Mr. Johnson is the president of NTERGAID, developers of a variety of hypertext and hypermedia tools. For the past several years he has been involved in the documentation, design, and document construction of the Black Magic, Help System II, and HyperWriter hypertext/hypermedia authoring systems.

The tutorial will focus on the following areas:

- Hypertext isn't a "Hyper-Mess"
- Hypertext documents have structure
- Analyzing hypertext documents for structure
- Structuring service & repair documents
- How the authoring tool affects document structure

"Hypermedia and Visual Technology"

Lloyd Walker

John Frassanito & Associates

Lloyd Walker is a Senior Designer at John Frassanito & Associates, an industrial design firm in the Clear Lake area that has specialized in conceptual, visionary planning for various offices at Johnson Space Center including Advanced Programs Office, Planet Surface Systems, Lunar Mars Programs Office, and Man/System Division. JF&A creates artwork and high level graphics to support the brainstorming, communication, and management of the complex issues associated with the space program.

The tutorial will focus on the following areas:

- *Hypermedia software as a visual database shell.*
- *Integration of images and text from a variety of sources.*
- *Simulation/Prototyping of graphical based control systems.*
- *Hypermedia based animations.*
- *Image acuisistion and creation.*

December 3, 1:30-5:00 p.m.

"Addressing Hypertext Design & Conversion Issues"

Robert J. Glushko

Search Technology, Inc.

Mr. Glushko is a Senior Staff Scientist at Search Technology of Norcross, Georgia, where he has worked since 1987 in the research, design, and development of hypertext concepts in design support, operational, and maintenance domains. He previously worked at the Software Engineering Institute at Carnegie-Mellon University and at AT&T Bell Laboratories.

The tutorial will focus on the following areas:

- Hypertext concepts: components, links, navigation, and entry points, session support
- Example applications and case studies
- Design guidelines for databases and user interfaces
- Project planning and management issues

"From HyperCard to Interactive Video"

Ellen Raghavan

Houston Community College System

Dr. Ellen Raghavan directs the dessktop publication, desktop presentation, and desktop production for the Houston Commununity College System. This tutorial will feature a hands on approach to hypermedia. The functions and potential of hypermdia and interactive video will be persented through various applications. Participants will have the opportunity to apply these methods.

The tutorial will focus on the following areas:

- Hands-on use of hypermedia applications
- Using color to create exciting hyperdocuments
- Converting one application to another
- Using interactive video as a communication/ traing tool
- Creating professional quality outputs

Key note Address

Hypertext, Typographic Man, and the Notion of Literacy

Patricia Ann Carlson

Session 1

State of the Practice

Chair: David Proctor

Evolution of the Experiment Document Information System

Jane Moorhead
Henry Brans

Engineering Software Development with HyperCard

Robert Darko

Life Sciences On-Line: A Study in Hypermedia Applications

Linda Christman
Nam Hoang

N91-21941

EXPERIMENT DOCUMENT INFORMATION SYSTEM (EDIS)

EVOLUTION

JANE MOORHEAD & HENRY BRANS

HOUSTON APPLIED LOGIC

HOUSTON, TEXAS

EXPERIMENT DOCUMENT INFORMATION SYSTEM (EDIS)

EVOLUTION

ABSTRACT

The EDIS is the second generation of a system designed to produce and control a document containing large amounts of text in combination with tables and graphs of mathematical/scientific data. The first generation system proved the concept, but the slow, unfriendly user interface resulted in an effort to find an off-the-shelf product to improve the interface capability while maintaining the system requirements. The basic design of that first system was combined with the hypertext concepts inherent in Hypercard to generate the much more usable EDIS. Currently in the latter stages of design, the EDIS promises to be the first step in the automation of the process required for defining complete packages of Life Sciences experiments for the Shuttle missions.

1. EXPERIMENT DOCUMENT AUTOMATION - BACKGROUND

The Life Sciences Experiment Document (ED), under control of the NASA/JSC Project Engineering Branch of the Life Sciences Project division describes a single experiment to be flown on a Shuttle Spacelab mission. It defines all functional objectives, inflight equipment, consumables, measurements, ground support, and test sessions, along with the expected results of the experiment. Preflight, postflight and training requirements also are a part of the ED. Working with Life Sciences engineering support, the Principal Investigator (PI) builds the ED using the "ED - Format and Instructions" document as a guide. The ED consists of sixteen chapters plus appendices. The fixed, or boilerplate, text contained in some sections of the ED applies to any Life Sciences experiment and references table formats that are completed by the PI with experiment-specific text and mathematical/scientific data. Other sections of the ED contain experiment text that is tailored for each experiment.

The ED is placed under configuration control in three phases, with about one third of the document placed under control following the experiment definition phase, another third placed under control at the Preliminary Design Review, and the final third placed under control at the Critical Design Review. One facility, the STI Center, is charged with the responsibility of documenting changes requested by the ED authors and approved by the Configuration Control Board.

One of the major difficulties of ED generation using only word processing methodologies was the manual update and control of pieces of related information occurring in multiple locations, and in the case of some numerical data embedded in algorithms or graphics. The LSPD requirement to provide effective and reliable PI support for an increasing number of experiments led to their decision to automate the processes for defining, modifying, and printing ED's.

2. AUTOMATED LIFE SCIENCES EXPERIMENT DOCUMENT (ALSED) - THE FIRST GENERATION

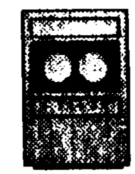
The initial automation of the ED and the experiment description was designed on a micro-VAX, using Datatrieve, Forms Management System (FMS), and FORTRAN. This system verified the capability of combining large amounts of text with mathematical data to produce an ED, while maintaining data integrity and configuration control. However, there were limitations to the system, including slow interactive response during execution of the data base management system interface and the requirement for all data to be entered via terminal access to a host computer. Additionally, and of no small importance, the system did not include a state-of-the art user interface.

3. EDIS - THE CURRENT GENERATION

Investigation of other ways to address the same requirements led to the choice of an implementation using Macintosh HyperCard. For the second generation of ED automation, the strengths of the ALSED system design were combined with a state-of-the-art user interface.

The EDIS will be implemented in three major phases (Figure 3-1). Phase 1, currently in progress, automates the PI experiment definition process, as well as some elements of the configuration control function. In Phase 2 the EDIS user training will be initiated and automation of the configuration control functions will be completed. Phase 3 will include implementation of a data base management capability, designed to store multiple ED's and to print specialty documents relating information from multiple experiment descriptions.

EDIS IMPLEMENTATION PHASES



	PI USER	CONFIGURATION CONTROL	LSPD DATA REPOSITORY
Phase 1	o define and modify experiment o print & "redline" ED o request for baseline change	o process "paper" CR o print formal ED	
Phase 2	o generate Change Request	o process and print Change Request	
Phase 3	o bulletin board	o print specialty documents	o "check out" & "return" experiment description o control experiment o control specialty data bases

Figure 3-1

Products for printing high quality documents for publication have not been selected. A final determination regarding whether to use a Macintosh based data base management capability, or to implement this final phase on a main frame is still under consideration. However, the plan is to use products that interface with HyperCard, avoiding interfaces requiring extensive application programming. The user help, training, and general EDIS information will be designed to resemble a "standard" Apple/Macintosh product, to take maximum advantage of user familiarity with personal computers.

4.0 EDIS DESIGN DESCRIPTION

The EDIS interface to the PI user consists of two major components: 1) the experiment description, and 2) the experiment document, or ED. In response to EDIS requests for information, the user will describe the experiment, and that information will be reflected in the tables and graphs within the document. The ED text will be available for viewing and for editing, also.

The experiment is provided to the PI user on a diskette, "checked out" of the STI Center. The home card contains buttons providing access to the experiment description and the "paper" ED, plus orientation and installation information.

4.1 Define And Modify Experiment

In the EDIS, an experiment description is pictured as a set of sessions comprised of activities requiring equipment, measurements, samples, etc. Definition of the experiment requires specification of all of these elements within a session. An element (e.g., activity, measurement, sample, equipment, hypothesis) is defined individually, and also contains references to other elements (i.e., an activity may reference equipment). These references are presented to the user as a list (e.g., a list of the equipment items defined for an activity). The user can add to the list, delete from the list, or move to a specific element on the list to view/edit the detailed information about that element. The user is prompted for all the information about all the elements required to generate the tables contained in the formal, printed ED.

4.2 ED Text

The "boilerplate" text is all text within the ED that remains relatively constant during the definition of the experiment. The "boilerplate" text is included on the user diskette when it is initially generated, and is determined by the type of experiment. Generation of the user diskette is a function of the configuration control process.

The ED tables of mathematical/scientific data are built from information entered by the user during the experiment definition. The user adds the current unbaselined experiment description to the ED tables by selecting the ED table update button on the EDIS main menu.

4.3 Configuration Control

Changes made to unbaselined elements of the experiment are incorporated immediately into the master ED when the diskette is returned to the EDIS configuration control administrator. Appropriate Change Notices and Change Pages are issued by the administrator, also.

The EDIS administrator builds the Formal Change Requests from the differences between the baseline copy of the experiment and the version submitted by the PI user. These requests are then submitted to the Configuration Control Board for formal approval. Following approval, the changes are implemented in the baseline experiment, and a copy can be returned to the PI user for continued inputs.

In Phase 3 of EDIS implementation, he PI user will be able to view experiment change status via a bulletin board facility.

5. EDIS - THE NEXT GENERATIONS

Following implementation of EDIS Phase 3, consideration will be given to providing a multimedia capability for including graphics and photographs in the ED. The EDIS logic will be expanded to aid the PI user in entering information specific to his particular experiment. The data base concept of storing and sharing experiment information may be extended to other aspects of LSPD mission definitions. Application of the EDIS concept to the Life Sciences requirements for Space Station will be investigated.

* * * * *

Acknowledgment: This work is supported by the Life Sciences Experiment Division, NASA/JSC.

N91-21942

Engineering Software Development with HyperCard

Robert J. Darko

Systems Programmer/Analyst

Hudson Products Corporation

A case study describing the successful and unsuccessful techniques used in the development of software using HyperCard. An evaluation of the viability of HyperCard for engineering and a discussion of the future use of HyperCard by this particular group of developers.

Engineering Software Development with HyperCard

Abstract:
A case study describing the successful and unsuccessful techniques used in the development of software using HyperCard. An evaluation of the viability of HyperCard for engineering and a discussion of the future use of HyperCard by this particular group of developers.

Introduction:
A brief description of Hudson Products Corporation is necessary to keep the rest of this paper in perspective. Hudson Products Corporation's major products are Air Cooled Heat Exchangers, Steam Condensers, and Fans. All of these products are designed in-house. Prior to the use of the Macintosh computer two and a half years ago, all of these calculations were done using batch software on a mini or mainframe computer or were done by hand. The majority of the software that existed was written in FORTRAN.

Much of the software that is used today had their beginnings in the 60's and 70's, and have gone through many transitions. The ideal solution was to rewrite the software and make it easier to maintain, but the decision to discontinue the use of the mini and mainframe computers as soon as possible was a major factor in some of our development decisions. Our first two concerns were the user interface and the linking of that interface to a particular software package.

The User Interface:
HyperCard was chosen as the development environment for the user interface. The ease and speed in which a particular interface could be created and altered was HyperCard's strongest point. Having chosen HyperCard, we set out on a user interface design. This seemed like an easy task, but it was soon discovered to be more difficult than had been anticipated. The engineers (as with most users) were set in their ways and initially were not open to a great deal of change.

For example, the three lines of text in figure 1 are a sample of the input that the engineers created prior to the use of HyperCard. Compare this to the screen depicted in figure 2 that is used today. Both figures reflect the same data.

```
BOGUS CHEMICAL CO.  H92-260-E       STEAM CONDENSER      SUMMER CASE    11/12/90nnc
11  900  120   4000      0   5    09 361 1      63600380400 300 0 088100     5    0
 1238  531250  710 3 19903123853   0    0 1238  531250  710 3 19903123853   00000
```

Figure 1.

When the development team came up with its initial design they assumed that the user would be comfortable with the point and click environment that the Macintosh provided. Feedback from the existing users showed a great desire to keep hand movement to a minimum, but the development team wanted to make the interface easy to use for experienced Macintosh users as well.

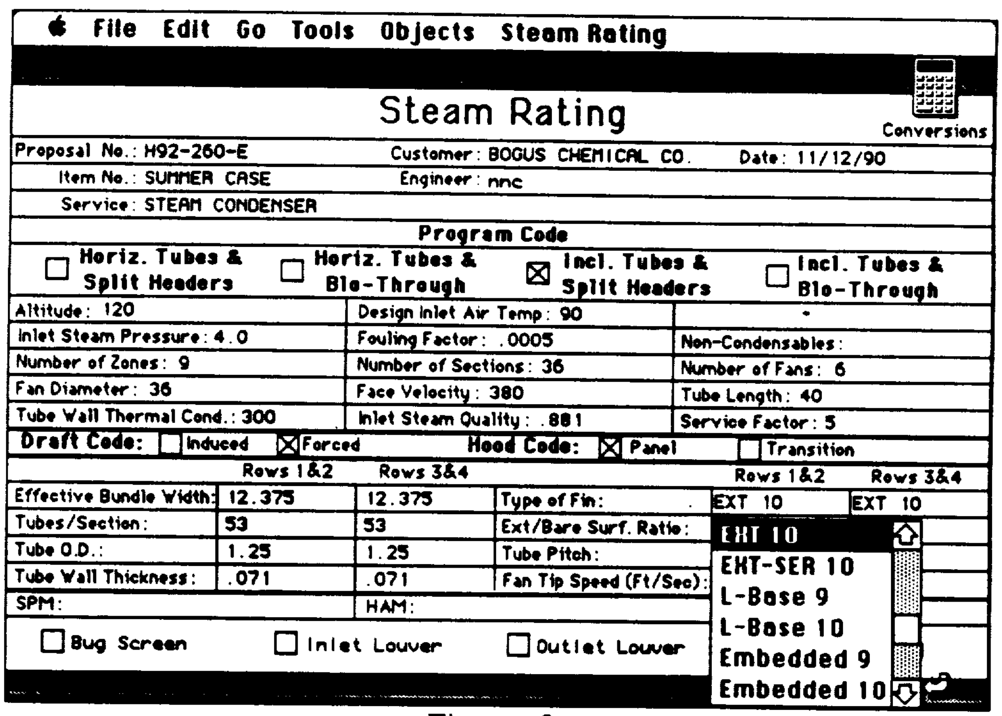


Figure 2.

The engineers wanted to use the mouse as little as possible, the development team wanted to make the data entry as fool-proof as possible. Consequently the use of pop-up menus was made when there was a limited choice for input. This required the user to move from the keyboard, to the mouse and back again. The solution was to produce some code that allowed the user to "tab into" a menu, or a "tabbable pop-up menu". An example of this is the scrollable list seen in figure 2. The bulk of the code that did this was in an XCMD, (an XCMD is basically a subroutine written in some high-level language and then physically linked to the HyperCard stack).

The data for this card is entered by using the tab key to go from field to field and typing in the data and then typing return or tab to go to the next field. A field is indicated by a label to the left of the field, the label will contain a colon so as to distinguish it from the data entered. Some fields, such as the one labeled "Type of Fin" in figure 2, would present a pop-up menu or list. The user could then use the arrow keys to change the selection and press tab or return to enter the data, or press the escape key to leave the entry unchanged. The different groups of buttons on this card represent different flags that were set in the original input.

The icon labeled "Conversions" is a button that provided a link to another stack that was developed by one of the engineers. The "Conversions" stack allowed the engineers to perform unit conversions to and from a variety of units. A secondary benefit provided by this stack is that the engineers now all use a common set of conversion factors.

After getting the feedback about the "Steam Rating" stack we went on to develop a stack for another program called "Check Rating". Figure 3 shows the card that was created. Again we used the "tabbable pop-up menu", but also made less use of buttons on this stack. By using our pop-up menus instead of buttons, most of the input was kept at the keyboard, where the user wanted it, but the user still had the ability to use the mouse in the usual way. The fields that have a pop-up menu have boldfaced labels, figure 3 shows an example of the menu for "Fin".

The user interface may look like a trivial problem on the surface, but we found a large portion of our time devoted to issues such as those mentioned above. HyperCard proved to be flexible enough to allow us try a variety of solutions to a given user interface problem, and thus pick a solution that fit our needs the best.

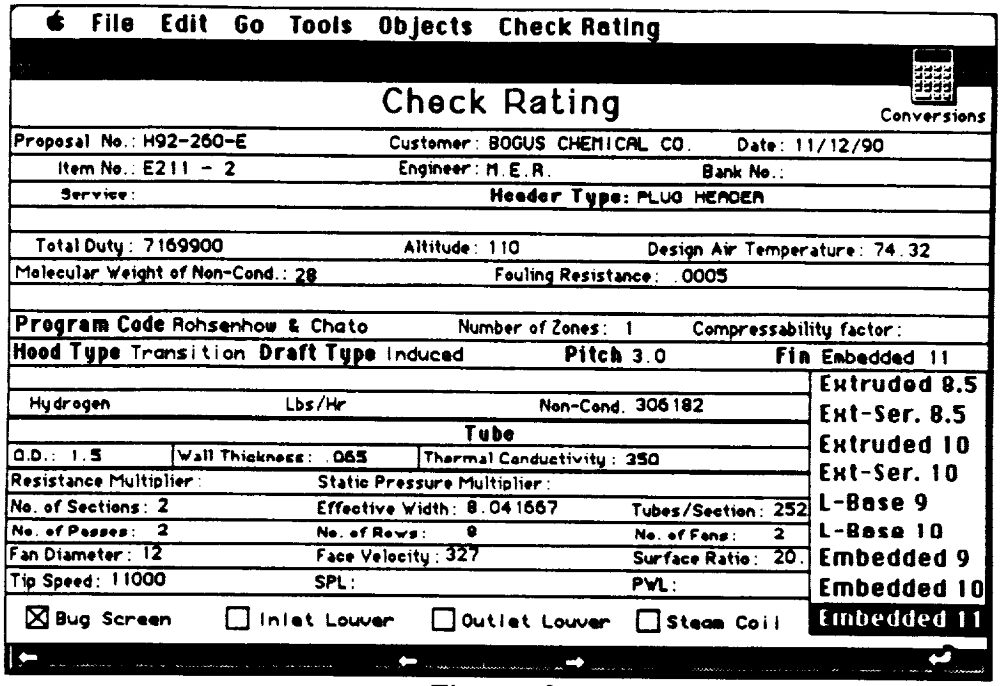


Figure 3.

Linking the Interface:

Getting the user interface linked to the code that does all of the real work was another problem. We had a series of programs that originally ran in a batch mode that were written in FORTRAN. HyperCard had no direct way of communicating with this code. Normally an XCMD would be written to handle a series of calculations that would be difficult to implement in HyperTalk, but the time necessary to rewrite the code in C or Pascal was too involved.

We decided to use a third party tool that allowed us to convert FORTRAN code into an XCMD. This tool involved installing an XCMD into every stack that needed to access the FORTRAN code. The stack could then invoke this XCMD and pass information indicating which FORTRAN routine to run. The XCMD contained a runtime environment for the FORTRAN which allowed HyperCard to pass information to the FORTRAN code and the FORTRAN code to pass information back.

This solution worked well at first, but several problems occurred. The first problem involved the implementation of the third party tool. It was not always consistent in the way that it located the FORTRAN subroutine to be run. The initial design required the user to have a folder that contained the stack and a subsequent folder that contained the FORTRAN code, as shown in figure 4, but due to the way that this tool worked the FORTRAN code had to be placed in the System folder instead.

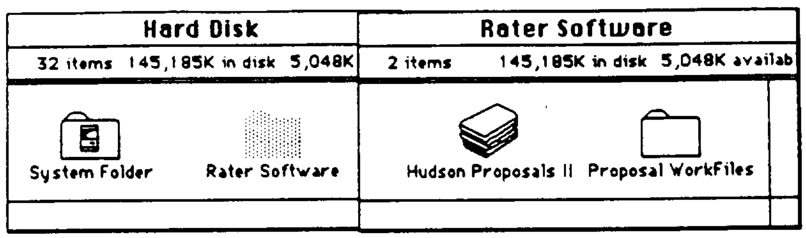


Figure 4.

The second problem with this solution was the time and memory overhead involved. The FORTRAN runtime package and subroutines required that at least 1.5 megabytes be allocated to the HyperCard program. This was acceptable, but not always desirable. The time overhead involved communication between HyperCard and the FORTRAN code. The third party XCMD situated itself between HyperCard and the FORTRAN and caused undesirable speed reduction in passing parameters back and forth. In fact we discovered that it was faster to write the parameters out to an external file and read them back in.

Several tests were conducted to solve each of these individual problems. The decision to make each piece of FORTRAN code a stand alone program seemed the best solution. This decision allowed us to revert to our original design of having the stack and a folder containing the FORTRAN programs in a directory of their own as shown in figure 4. This made software updates easier to manage since the user could copy the contents of one directory to his disk and not worry about putting different files in different locations.

Our communication mechanism was simple. HyperCard would gather input from the user, write that data out to a file on disk, and then invoke the FORTRAN program. The FORTRAN program would read in the data, calculate the results, and write them out to another file, which would be read in by HyperCard and then reported to the user. All of this worked quickly enough to satisfy our users.

Other Concerns:

Another concern after getting the rudimentary portion of our stacks to work, was the need to store and fetch data between runs. We needed a database management system, but that area was still being studied and a complete solution had not been decided on, so a temporary solution was devised.

The easy solution was to save a copy of each stack after the input had been made, but there was a lot of overhead in the stacks. In fact the stacks themselves had been merged into a single stack which totaled more than 250 Kilobytes. Saving this much information was going to exhaust our storage rather quickly. The size of the data that actually needed to be stored was around 5 Kilobytes. So a scheme was devised that allowed us to store and retrieve the data in a text file. This technique required that the card design not change drastically. The most important restriction was that the deletion of fields or buttons was not allowed. We could add some if needed, but could not delete them and still be backwards compatible with our data format.

Using the Hierarchical File System of the Macintosh we achieved a pseudo-database that allowed our users to fetch related data. This worked by putting related files into the same folder, (see figure 5 for an example of this hierarchy). This technique lasted for about a month before two other problems became apparent.

The first problem involved the time needed to read and write the data files. Several solutions were tried, but the only one that achieved any appreciable time reduction was an XCMD. So we coded an XCMD to collect the data in memory and then dump it out to the disk. The drawback to this was that if we ever added new fields or buttons, then we had to recode the XCMD. This was acceptable.

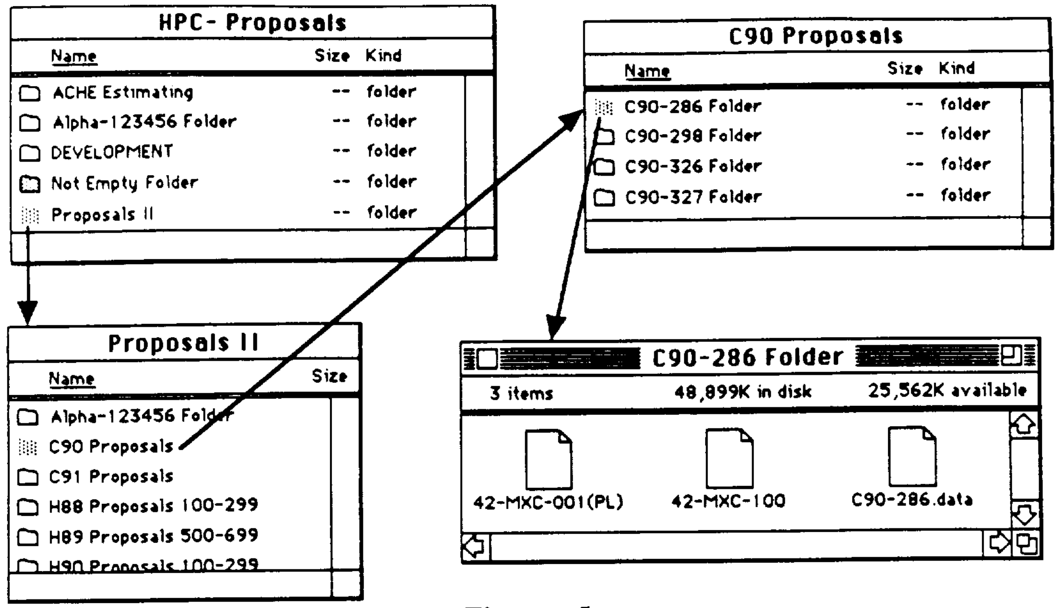


Figure 5.

The second problem became two-fold. Users were saving data on each of their individual systems. This sometimes led to a duplication of information when more than one engineer worked on the same problem. The solution here was to modify the stack so that it could use a file server instead of the user's hard disk. This was accomplished without a significant amount of effort, but within 3 months we hit another barrier. The number of files created was getting rather large. Since this was a temporary solution to begin with, we simply added another level of hierarchy to the storage of the data.

Printing was a problem that was handled two ways. The first was for simple output, the solution was an XCMD that printed multi-page data in a single font. The second was for more complex output. We needed to create detailed forms and fill in the form with data from the HyperCard stack. For this we turned to a third party product called "Reports!". The initial version did the job, but was a bit difficult to implement, this product has been revised now and is well worth using.

Future Goals:

We have already started to implement an SQL database that HyperCard will access. This will allow us to link not only our engineering data, but our accounting, scheduling, and statistical data as well. Other plans include on-line help systems for each of the stacks, simple expert systems, and recoding some of our software into true XCMDs.

Evaluation:

HyperCard has its shortcomings, namely speed and lack of color. The speed issue can usually be worked around by use of a compiler or an XCMD. Color is a problem that will have to wait until Apple does something about it. Fortunately our needs do not necessitate the use of color. Card dimensions had caused us some small problems, but they too can be worked around.

HyperCard's biggest benefit is its ease of creating a user interface. Since it is easy to make modifications to the card design many different solutions to a particular problem can be evaluated. We have also been able to create a series of tools that all have the same "feel" to the user. This has helped keep the need for user training down significantly.

HyperCard has allowed us to accomplish more than we could have using conventional development methods. It has proven to be an excellent prototyping tool, and a more than adequate environment for the implementation of our software. Due to the success with this project, new projects are being implemented using HyperCard and the techniques described.

LIFE SCIENCES ON-LINE:
A STUDY IN HYPERMEDIA APPLICATION

Life Sciences Project Division, NASA, Johnson Space Center

by

Linda A. Christman
Nam V. Hoang
David R. Proctor

N91-21943

Life Sciences On-Line: A Study in Hypermedia Application

by Linda A. Christman, Nam V. Hoang, GE Government Services, Houston, TX, and David R. Proctor, NASA, Life Sciences Project Division, Johnson Space Center

ABSTRACT

The purpose of this study was to determine the feasibility of using a computer-based interactive information recall module for the Life Sciences Project Division (LSPD) at NASA, Johnson Space Center. LSPD personnel prepare payload experiments to test and monitor physiological functions in zero gravity. Training refreshers and other types of online help are needed to support personnel in their tasks during mission testing and in flight. This paper presents results of a survey of other hypermedia and multimedia developers and lessons learned by the developers of the LSPD prototype module. The paper also discusses related issues and future applications as well as recommends further hypermedia development within the LSPD.

Life Sciences Research

Rapidly emerging media-based technologies are providing online information aids for professional and technical workers in both the public and private sectors. At NASA-Johnson Space Center, attention is focusing on these tools for space program use. The Life Sciences Project Division (LSPD) is researching the application of online "job performance assistance" (Dona Erb, August 1987) to Life Sciences flight support personnel.

Life Sciences astronauts perform experiments for inflight missions on the Shuttle and planned Space Station *Freedom*. These investigations use special equipment modified for space flight to study human physiological and biological responses that occur in zero gravity.

Within the LSPD there is a perceived need for online, context-sensitive help such as training simulations or quick reference information during mission testing. In addition, a need exists for similar online reference tools for personnel working in other areas of the LSPD: development and testing, science monitoring, technical lab activities, office automation, and administrative tasks.

Purpose of Study

In response to these needs, an online prototype module, Life Sciences Interactive Information Recall (LSIIR), was developed. Interactive media was incorporated into a computer desktop workstation environment with which mission or payload specialist, scientist or engineer, and support or administrative personnel were already familiar. The aim of this approach was to focus the viewer's attention on the screen and manipulate content in such a way as to allow instant information transfer from screen to viewer.

The purpose of the study was to demonstrate the feasibility of providing an online interactive information recall system within the LSPD.

Hypermedia/Multimedia Tools

Hypermedia is the extension of hypertext, a term coined by Ted Nelson (Nelson, Literary Machines, 1987). Hypertext is non-sequentially, electronically stored writing. For example, a key word or phrase in a document can be highlighted on screen and "linked" directly to a picture -- or another document. This link gives the user quick access to more information, without flipping through a paper manual, going to a library, or consulting a human source.

Online documentation and online help are not new concepts but context-sensitive linking and nonlinear selection are new capabilities of the emerging online information technologies. In addition, multimedia enhancements, such as video, 3-D simulations, voice, and animated text and graphics raise hypermedia to another level of training methodology. Hence, hypermedia combines information from different sources into one source, the computer.

These tools are now being used to develop appropriate online information and training media. The purpose of this approach is not to replace hands-on training methods or paper reference manuals but to amplify and augment these methods and materials electronically. The rationale for a multimedia-enhanced online system is that it increases visual information and allows the user to access all the information he needs on demand.

Issues

The new role of hypermedia as a learning concept and training tool raises some interesting and provocative issues:

Is a hypermedia/multimedia-enhanced system necessary? Hypermedia can provide online information without the addition of any multimedia resources. However, multiple media may provide the best presentation for topics that require the realism of photographs, the symbolism of diagrams, the simulation of operations, or verbal instructions.

Will online documents replace paper? The user's dependency on paper documents decreases as online documentation increases. Problems such as storing and updating paper documents, delays and inconvenience in document pick-up, or delivering too much or too little documentation are solved. Related materials no longer need to be limited to paper or stored and accessed separately from paper. Other media can be accessed with the document and appear as a window on the same screen.

What are its applications to education? The applications in education are well documented. An educational delivery system firm, Advanced Educational Concepts, used multimedia enhancements to convert a standard training area to a "projection" room by incorporating 3-D simulations, holographic projections, and interactive video. The system showed that students retain 25% of what they hear, 45% of what they hear and see, and 70% of what they do. Students in the IRIS project at Brown University reported that Intermedia (hypertext online referencing course) reinforced lectures and taught things not found in other sources. The EXVIS project at the University of Lowell recommended the use of icons with sound as well as visual parameters to represent several dimensions. This technique improved the performance of test subjects in analyzing scientific data.

Is it applicable to aerospace training? In the coming Space Station Era, information recall will no longer be limited to two or three day trips in space. Astronauts may be spending as many as 180 days in space. A simple pocket card or brief reference manual will not suffice. Storage space for larger documents may be unavailable. Messages and some instructions can be transmitted, but other information must be accessible from onboard systems. In another application, *Mechanical Engineering* magazine recently noted that a Macintosh running LabView 2 was used to pinpoint a fuel leak on the Space Shuttle *Columbia*.

Do the benefits outweigh the costs? How much user and developer time is saved? A pressing question is to what degree benefits justify development costs. Upgrades in multimedia hardware will speed and ease the tasks of the developer and user. Much of the software needed to develop an online training or information recall system is already available off the shelf, which means an online system does not have to be developed from scratch. Many HyperCard scripts (HyperTalk) are available through third party developers or shareware and can be used as is or with slight modifications for a new application. Many authoring packages, such as HyperCard, Macromind Director, and Guide have a low learning curve, at least to master the basics. Also, application updates like HyperCard 2.0 have added or improved features which can cut the developer's time in half and speed up processing and access time. Some types of training and information recall activities, such as simulations and trouble-shooting, may require a certain degree of user interactivity. Programming expertise may be required to implement some of them; however, others can be easily presented using interactive video. Multimedia hardware in the high-end range is now consumer-priced and therefore no longer cost-prohibitive for educators and contractors. The payback for development time and labor depends upon the amount of information required and selected for electronic access and the speed of information retrieval that can be built into an online system. In both aerospace and education, an easy updating procedure is important to keep content current. Additionally, in aerospace applications, a fast turnaround time for retrieving information is essential. If an online system can meet these requirements, it will pay for itself.

Objectives

The objectives of the study were: (1) to survey other users of hypermedia and multimedia tools in developing online training and information help systems for their employees or customers; (2) to produce a module (HyperCard stack) to serve as a training and memory aid that would run on a Mac SE or SE/30; (3) to review the end product (module) and evaluate its possible application to crew training or other types of online training within the LSPD.

Stack Development

The development team consisted of an instructional designer and a development programmer using two Mac's: an SE and an SE/30. A college student assisted in creating and modifying the graphics and sounds in the stack. Other resources outside the team were called upon as the need arose. Development of the stack extended from February to June 1990. The project was completed on time, including several modifications made as a result of the preliminary review. The stack was presented to LSPD and contractor personnel for review.

Hardware used in development included the Macintosh SE and SE/30 with an 80 Mb hard drive. The Mac SE's were connected to the Macintosh network to enable use of online utilities and provide copies to reviewers. This setup insured stack protection and and developer independence in producing segments of the stack. Software included: HyperCard 1.2.2 /1.2.5, the MacRecorder Sound System, MacDraw, MacPaint, Canvas, and Macromind Director. Both clip art and scanned photos were used; an Apple scanner was used to scan photos.

Content

The study focused on the production and demonstration of a prototype stack for refreshing the training and knowledge of LSPD users in the operation of an inflight instrument. The instrument was a flow cytometer used in a hypothetical payload experiment to measure cells in zero gravity.

The operation of the flow cytometer was divided into several sequential segments: (1) Introduction, (2) Equipment, (3) Procedure, (4) Configuration, and (5) Operation. The stack demonstrated a generic sampling of content vs. specific requirements for the purpose of the study. Procedures and operations were based on real-life experiments and considered to be representative of a typical usage of the inflight flow cytometer.

Concept

The flow cytometer was considered a good choice for creating a simulation of the operation for online viewing. It allows the user to step through the operation visually without using the real equipment. He can see the results of possible effects. In the event of an incorrect selection, the stack would branch to the correct action required. The stack "coaches" the user without having to set up the real experiment. Further, it "immerses" the user in the experiment without having to take time out to reference the hard copy documentation. Hence, protocol correctness can be ensured with online help and information.

Design

An "online help" stack was envisioned that would cover all the critical information a user might need during testing or in flight. It was assumed that (1) the training task involved more than seven items to be recalled, (2) the training protocols could be changed, and (3) testing and feedback reinforce learning and reduce errors in performance.

Simulation and walk-through sequences were incorporated into the stack. A storyboard was drafted to show the sequencing of content based on the flow cytometer used in flight. The original screen maps were used as building blocks which changed as the stack began to take form. A user navigation map was added later (see USER NAVIGATION MAP on the next page). Enhancements based on design modifications and preliminary review removed some of the "flat" storyboard quality of the finished product. These enhancements were created primarily with HyperCard and MacRecorder. Macromind Director was used to "smooth out" the opening and closing screen animations. Graphic windows were created as placeholders for scanned photos or video clips using tools that would be acquired later. Voice overlays and other sound and graphic segments were modified to simplify or clarify the user interface.

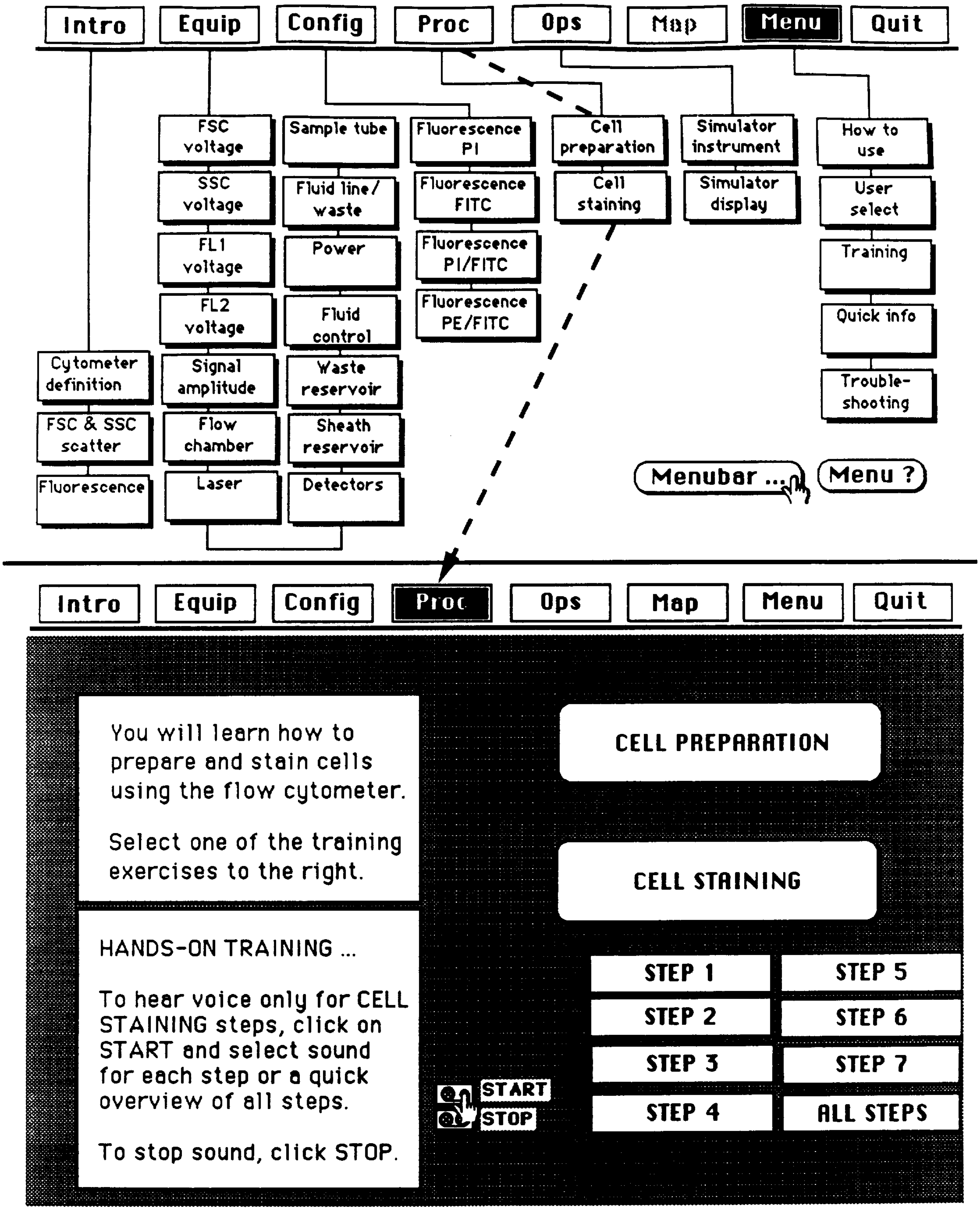
Since the end product was delivered to Mac SE users for review, still video and camera enhancements would have displayed with a very low resolution at best on the normal Mac SE black-and-white monitor. Full motion video and 3-D graphics are more effective on a Mac II or computers with compatible platforms.

Modes

The software selected to create the stack, HyperCard, is an authoring toolkit primarily used for training and presentation applications. It allows the author to create and run a program, or stack, in several modes: programmer and user. While the stack is being created, the author is functioning not only as author but also as user so that every segment constructed can be tested, as well as modified, at the user level. There are five user levels: browsing, typing, painting, authoring, and scripting.

Programmer Mode

In this mode, the programmer authors and edits the stack using the authoring or scripting user levels. The authoring level is selected to link one card in the stack to another card, usually in sequence. At this level, scripting cannot be done using the built-in HyperTalk scripting



Example "Link" from User Navigation Map

language (similar to but simpler than higher level languages such as Pascal or 'C'). At the scripting level, the programmer can use HyperTalk 's English-like functions which allow rapid cross-linking of information segments. At this level, the author-user can use the features of HyperCard to their fullest potential.

Normally, a stack is tested by the user in the run-time mode at the browsing level. In the case of the flow cytometer stack, the user level was set at the typing level because the type command was used to create running type, as in the trouble-shooting exercise, which provides user prompts and feedback.

User Mode

In the run-time mode, the stack is password-protected inside the stack. The password protection ensures that, at the browsing, typing, or painting levels, the user can navigate through the program but cannot access or change any of the HyperTalk scripts that control the links between the cards in the stack. If a user wishes to view or edit the scripts, he must select the Apple command key to get the full menus, pull down the File menu, select Protect Stack, enter the password, and select the Scripting user level in the dialog box. At the Painting level or above, the user can alter any of the graphics created or imported.

Interactivity

The mouse was used as the primary input device. Keyboard input was required in the data acquisition segment (Operation simulation) only if the user needed to change parameters after answering the user prompts. Various levels of interactivity were built into the stack. At lower levels, information is retrieved with minimal participation from the viewer. At higher, more complex levels, such as training simulations and procedural steps, more user control should be provided. The developer's selection of a particular level should be determined by the user's purpose and requirements, such as learning , reviewing , or recalling the material.

Size

The run-time stack at the end of the project contained 1.7 Mb. It was determined that graphics, animations, and sound added to the module's size along with scripting and deletions. Compacting the stack removed extra space and helped speed up navigation. The stack also varied in size when the Trouble-shooting exercise was accessed by the user. A copy of the stack was placed in a folder with HyperCard and Macromind applications on the network server for review.

Approach

The approach used to set up the stack for computer screen viewing was to provide a two-way "task mirror" that would both reflect the task being performed and reveal the information the user needs to accomplish the task. With these options the user would be able to (1) step through the simulated task in a training session, or (2) quickly recall the specific items needed to complete or troubleshoot the task. This open-ended approach provides a refresher of previously mastered information, updates old information, and allows the user to learn less prioritized or infrequently performed tasks.

Hence, the computer serves as a "trainer" or simulator. It provides the user with different sets of information to change variables during a training exercise or make alterations to procedures and configurations. The user can point and click the mouse cursor on these various sets of information, via icons (buttons, pictures, or text), and link instantly to another desired set of information. It is similar to flipping to the index or table of contents and then to the first page of a section in a paper document, then flipping again to the specific page, and finally using eye and finger to move down the page and locate the lines that contain the information needed. In comparison, online linking takes less time.

Format

Three user paths (USER MENU) link the user to online training and information: Training Session, Quick Info, and Trouble-shooting. "Training Session" provides a full, walk-through session for a first-time viewer. "Quick Info" allows the user to find information on demand without going through any learning exercises. "Trouble-shooting," represents a knowledge base of built-in maintenance and repair information in the event of a malfunction or in cases where the user needs to change procedures or configurations.

A list of information segments presented in a hierarchical order was added at the top of the screen (MENU BAR) for user navigation through the stack. The menu bar also contains a navigation flow chart (USER MAP) showing the location of the user within the stack. The icons on both the menu bar and the user map are clicked to go to any training activity or information set desired. Hence, the navigation map allows the user to browse or search through a linear sequence of information -- non-sequentially -- from anywhere in the stack.

Consistency was maintained in the placement and type of icons and buttons used to link to other cards in order to reduce user "slow-down" in navigating through the stack.

Implementation Considerations

Many factors can affect the implementation of online modules within the LSPD, such as:

- Portable vs. fixed computers for workstations
- Cordless input devices (i.e. touch screen and voice command) vs. the mouse and keyboard
- Computer access time related to user needs
- Trainer or expert input on training protocols and information requirements
- User feedback on training and information needs

Hypermedia requires further investigation within the LSPD as a computer-centered medium for a user information workstation. Results of providing information on line in ground testing situations would be aimed toward the Space Station Era.

Specific inflight implementation problems that should be addressed are:

- Effect of visual disorientation on computer screen proportions
- Effect of light conditions and viewing angles on computer displays
- Distortions in computer sound and voice command
- Onboard storage vs. transmission of online information
- Use of online assistance as a stand-alone system or with paper references

Conclusions

The flow cytometry prototype module demonstrated how an interactive information recall module could be applied in a scientific and engineering work environment to refresh knowledge acquired in training or provide an orientation to a new task. A hypermedia approach to development provided the best user interface and quickest access to information needed. The integration of multimedia, including video, animations, 3-D simulations, knowledge bases, sound, and more interactive input devices could not only increase visual information by transforming and enhancing online displays but decrease time spent in review.

Evaluators recommended the stack for further development and rated features such as the user interface (navigation map), graphic zoom-in's, and walk-through steps (without the voice) highest in user value.

Survey Results

A survey was conducted to determine the type of hypermedia/multimedia-based online training and information retrieval projects which are in the planning stages or under development by other individuals or organizations. Respondents were mostly from the Southwestern U.S. Several participants also provided other contacts who were involved in similar projects. Approximately half of the respondents were currently working on a hyper/multimedia project. Most products were used to train or provide online help and job support or quick information interchange. Most projects were ongoing; independent development lagged behind commercial development. Target users included company personnel, management staff, NASA, medical staff, educators, and researchers. Both Macintoshes and IBM's as well as mainframes were used in development. HyperCard, SuperCard, Macromind Director, Guide, APDA (Apple Programmer Development Association) products, and desktop publishing applications were used. Most respondents were or would be integrating expert systems. Most were willing to demonstrate their products and participate in a hypermedia work group.

Lessons Learned

Numerous design and user interface changes were made to enhance the stack. In addition, HyperCard 1.2.2 and 1.2.5 were lacking in features that would have improved the stack. The main lessons learned from stack development are described below.

- Macromind Director enhances graphic displays and animation. Director also shows a shift in data displays when parameters are changed by the user.
- Selecting scenarios to show a sequence of location or events from the actual work environment add real-life perspective to graphics.
- Sound can be turned on or off.
- Voice should be used sparingly to emphasize critical information or to give warnings, or in cases where the work or training area is not adjacent to the computer.
- Short sound clips are more effective than long soundtracks (music).
- Special effects focus user attention when a change occurs.
- Successive steps on the screen can occur automatically or be user-selected.
- A navigation map should be placed in the background of the stack, not on the cards.
- Mac SE processes large HyperCard stacks more slowly than the Mac SE/30 or Mac II.
- HyperCard is ideal for creating storyboards, such as creating placeholder "windows," stick figures, and pseudo animation for video and simulations.
- For running type to work in HyperCard (as in the Trouble-shooting exercise), the user level for the run-time program must be set to Typing.

- Setting a password and the user level to Browsing or Typing to prevent the user from changing screen objects or scripts (vs. the more dangerous practice of protecting the stack and locking out the developer as well as the user).
- Writing a script to automatically compact a stack. Compacting a stack eliminates free space (the number of bytes added to the stack when a new action is performed). This saves disk space.
- Shortcuts in scripting (thanks to a reduction of the developers' learning curve and HyperCard version changes) were responsible for speeding up online information access time:

  - "Send mouse up to card button..." automatically points and clicks.
  - Visual effects work in color mode on Mac II in HyperCard 2.0.
  - Characters in text fields can now be underlined, bolded, resized, and restyled individually within the field to make sections like the Introduction more readable.
  - "Set hilite to true/false" for each object to stay highlighted until another button was clicked is now a button function called "shared hilite."
  - "Put [literal or variable] into card field of card" does not have to be repeated because text field can be put in the background of the stack, not on the cards.

These new features can save a substantial amount of time spent in scripting and thus speed up processing and information access. One example is creating one background card and one line of 'set' and one line of 'put' instead of multiple lines for a set of cards with minimal changes resulting from user interactivity. This type of activity occurs in the Operation and Acquisition segments of the flow cytometer training simulation.

<u>Recommendations</u>

The LSIIR hypermedia prototype in this study proved a concept for an online help system that could not only save time and error when critical information is needed but also decrease turnaround time in obtaining information and increase the performer's productivity.

The users who would most benefit from a hypermedia-based system are LSPD mission end users, such as astronauts and PI's, and LSPD engineers and scientists. Other groups that could benefit from such a system would be contractors, support personnel, and visitors to the LSPD.

Ongoing development should include online systems for:

- Electronic documentation and information
- Electronic training and review

Module concepts that derive from this research include a desktop referencing system as well as an online help system.

A desktop referencing system would include an LSPD module with a "Q & A" display version for visitors and a more comprehensive version for LSPD personnel; an online LSPD guide of the branches: SE, SE2, SE3, SE4; and a search and retrieval front-end program for viewing documentation on line. An online help system would continue development of training and review modules for flight support personnel, such as a Flow Cytometry series, including a FlowCytoMETRIC module, which instructs the user in how to calibrate the flow cytometer based on different sets and tests and a FlowCytoEXPERT module which provides the user with a knowledge base for trouble-shooting the flow cytometry equipment ("stacks-in-racks"). Instructions in analyzing data, reading histograms, and explaining technical symbols and expressions for technical monitors and principal investigators could also be put on line.

Future Research

Research that should be continued or initiated within the LSPD includes:

- Integration of expert systems into online help systems
- Indentification of state-of-the-art multimedia technologies
- Cost/benefit of online vs. paper documentation
- Evaluation of training and information needs
- Research vs. non-research tasks performed in flight
- Study of flight and payload training requirements
- Investigation of other hypermedia/multimedia research

References

Beeman, William O. et al., "Hypertext and Pluralism: From Lineal to Nonlineal Thinking," (IRIS Project), Hypertext '87 Papers, 1987

Betts, Kellyn S., "Macintosh Seeks Engineering Muscle," Mechanical Engineering, vol.112, No. 7, November 1990

Erb, Dona M., MITRE Working Papers, The MITRE Corporation, August 1987-June 1989

Nelson, Theodor H., Literary Machines, ed 87.1, The Distributors, South Bend, Indiana, 1987

Rice, J.W., survey respondent, Advanced Educational Concepts, Inc., Houston, Texas

Williams, Marian G. et al., "Computer-Human Interface Issues in the Design of an Intelligent Workstation for Scientific Visualization," (EXVIS Project), SIGCHI Bulletin, vol. 21, No. 4, April 1990

Session 2

Education and Training: Directions in Hypermedia

Co-Chair Glenn B. Freedman
Co-Chair Glen Van Zandt

Computer Assisted Knowledge Acquisition for Hypermedia Systems

Kurt Steuck

A Knowledge Based Browser Using Hypermedia

Tony Pockington
Lui Wang

A Model for Addressing Navigation Limitations and Metacognitive Constraints in Hypermedia Training Systems

Glenn B. Freedman

N91-21944

COMPUTER-ASSISTED KNOWLEDGE ACQUISITION FOR HYPERMEDIA SYSTEMS

Kurt Steuck
Air Force Human Resources Laboratory

Abstract

The majority of this paper describes how procedural and declarative knowledge can be used to set up the structure or "web" of a hypermedia environment. The Air Force Human Resources Laboratory (AFHRL) has developed an automated knowledge acquisition tool (AKAT) that helps a knowledge engineer elicit and represent an expert's knowledge involved in performing procedural tasks. The tool represents both procedural and prerequisite, declarative knowledge that supports each activity performed by the expert. This knowledge is output and subsequently read by a hypertext scripting language to generate the links between blank, but labeled cards. Each step of the expert's activity and each piece of supporting declarative knowledge is set up as an empty node. An instructional developer can then enter detailed instructional material concerning each step and declarative knowledge into these empty nodes. The paper ends by describing other research that facilitates the translation of knowledge from one form into a form more readily useable by computerized systems. (Slide 2.)

Background

The Intelligent Systems Branch of the Training Systems Division of the Air Force Human Resources Laboratory (AFHRL) is continuing the research and development of intelligent training technologies. We investigate the application of artificial intelligence principles to compter-based training to produce training systems that behave intelligently. We have on-going efforts in the development of intelligent tutors in the operation, maintenance, and repair of complex physical devices, such as console operations and maintenance of the leading edge F-15 wing. We also have efforts which explore the application of artificial neural networks and other machine learning technologies to the modeling of a student as he or she progesses through training. A new effort will explore the application of virtual reality technologies to technical training. (Slide 3.)

Problem

Knowledge is available to instructional developers in forms that cannnot immediately be use for training or job-aiding. That knowledge is in our expert's heads, in paper form in technical documents and procedural guides, and in schematics. The problem is that we need to get that knowledge in a form that we can use in computer-based instruction, hypermedia, expert systems. We need to be able to translate, convert, or transform the knowledge from one form to a more readily useable form efficiently. Our approach in addressing this problem is to design and develop computerized tools that help us reconstruct existing knowledge to be used by computerized training systems. (Slide 4.)

The goal of this paper is to describe related technologies that convert knowledge form one form into different end products. One technology helps a training developer elicit and represent an expert's knowledge in performaing procedural tasks. The output of this tool can then be feed into a hypermedia environment to structure the "web" of information in much the same way as the expert described it to the training developer. The same knowledge can be feed into the second technology for use in other forms of job-aiding and training systems. The second technology is a natural language processor that takes procedural knowledge existing in technical manuals and converts it into expert system rules. To date, this sytem has been implemted in electronic domains to generate CLIPS rules.

Automated Knowledge Acquisition

Our first scenario is one in which critical domain knowledge resides in an expert, but is needed in for the development of an intelligent tutoring system or some other form of computer-based training system. Our approach is to design and develop computerized KA tools which automate knowledge elicitation, formalization, and representation processes. These tools are used to elicit knowledge from subject matter experts (SMEs) in analyzing or decomposing difficult tasks in the expert's domain. (Slide 5.)

The set of techniques used to elicit knowledge from a SME is referred to as knowledge acquisition (KA) or knowledge engineering. In KA for the development of training systems, a knowledge engineer, usually an instructional or cognitive psychologist, verbally elicits important domain knowledge from a subject matter expert (SME). This is a time consuming and difficult process, because the knowledge engineer typically is not intimately familiar with the problems, terminology, or problem solving techniques in the subject domain. Furthermore, the SME may experience difficulty in verbalizing relevant domain knowledge due to a lack of experience at describing the domain, the unavailability of proceduralized knowledge, or low motivation.

The objective of the Automated Knowledge Acquisition (AKA) effort is to design and develop tools which automate knowledge elicitation and representation. Successful automation would allow a SME and knowledge engineer to rapidly enter information directly into the computer without the aid of a computer programmer. A well designed interface would allow for authors to possess minimal computer skills and reduce some of the problems encountered by an expert and knowledge engineer during KA activities. The benefits of automating the representation of an expert's problem solving activities, events, and supporting information would reduce the time and resources required for KA. For instance, paper documentation is eliminated, formalization of the knowledge is less intensive, communication of the information to a computer programmer is not needed, and it does not require SME to fit his/her knowledge into a form constrained by the knowledge engineer.

Hypermedia Automated Knowledge Structure Acquisition Tool

The first technolgy to be described allows us to elicit procedural knowledge from an expert and represent it as the structure of a hypermedia system. (Slide 6.)

Overview of Steps

1. Start with classic expert scenario: the expert has years of experience, his/her knowledge is not documented, he/she may not be able to train many novices, and he/she may not be able to verbalize the domain information very well.

2. Together the knowledge engineer and the domain expert build a representation of the steps, subsequent events, and the supporting knowledge required to accomplish each of the steps.

3. The task description and the knowledge required to perform each step of the task is output in a frame representation.

4. The next step is to read the frames into the hypermedia system with a scripting language inherent in the hypermedia package. In our example, we used Hyperpad and its scripting language. We did not need to write any external code, such as C or Pascal, to accomplish the translation of the AKAT output into the hypermedia web. The knowledge represented in the procedural structure is reconstructed as the structure of the web. It is not the content per se that is important, but rather the structure of the knowledge. The scripting language was written to produce labeled, blank nodes or cards and their inks to next step or event and the prerequisite knowledge elicited in AKAT.

5. The training developer then enters more detailed instruction into the nodes for presentation to domain novices. A training developer then can interview the expert to fill in the cards with details of each step in the procedure.

Each of these steps will be covered in more detail in the next set of slides.

Automated Knowledge Acquisition (Interface)

The interface uses icons to represent different aspects of an expert's procedural knowledge and the scenarios or events that result when the expert performs each step. As an example one step is "Determine Impact on Load and/or Execute Plans" and the subsequent events are "Impact on Load Only," "Impact on Execute Only," "Impact on Both." (Slide 7.)

How is this done? The SME can create, move, or delete icons to represent the problem solving steps and the resulting events using a mouse and keyboard. The user creates this procedural net in a manner that reflects his/her own methods of completing the targeted task. After representing the procedures, the SME and KE fill in a node editor prerequisite knowledge related to each step. The SME can add prerequisite knowledge consisting of concepts, facts, procedures, and rules. For example, two concepts required in order to correctly perform the "Determine Impact on Plans" step are "Load Plans" and "Execute Plans." (Slide 8.)

AKAT Frame Output

After domain expertise has been elicited and verified, the knowledge can be output in a frame representation. The frame includes the name of the step, its children (i.e., the resulting events), and all of the related facts, concepts, rules, and procedures. (Slide 9.)

Hypermedia Knowledge Structure

The information contained in this frame is read by the scripting language to form the structure of the hypermedia web. It is not the content of any one node, but rather a set of nodes. One node is set up for each piece of information. One for each step, one for each of the resulting events, and so on. The links between the nodes are based on the category or type of information related to the node. (Slide 10.)

For example, the node "Determine Impact on Plans" has links to the events "Impact on Load Plans Only," Impact on Execute Plans Only," and "Impact on Both." Links to other nodes are based on the related concepts, facts, rules, and principles.

Hypermedia Nodes

Each node formed by the scripting language has several parts automatically created. Each node has a title derived from the name of the procedural step, links to the events, and links to supporting knowledge. Each node is initially blank. The training developer and omain expert can then systematically develop detailed instruction for each node. (Slide 11.)

The result of all this is a hyper environment that can be used for initial training or refresher training. A student can navigate (I had to use the word at least once in this paper) the web for procedural content or procede in a very comprehensive manner mixing visits to nodes containing procedural and supporting knowledge.

Expanded Approach

The AKAT to hypermedia translation is not the only approach we have taken in addressing the problem of having knowledge in different formats. AKAT was initially developed for converting procedural and supporting knowledge into LISP code for direct import into an intelligent tutoring system. This capability was designed to use the knowledge as part of the expert module of the ITS. We later, however, realized that we could dump out the knowledge in rule format. The preceding event set up the conditions for which a step was to be performed. This was easily represented as "IF <an event was true>, THEN <perform the step>." We modified AKAT to reproduce the net of steps and events as a set of IF <event>, Then <step> rules. (Slide 12.)

More importantly, another technology developed for transforming knowledge is a system called the Textual Automated Reduction System (TARS). The goal of the TARS effort was to develop a natural language processor that would read technical manuals (tech orders, TOs) and produce well-formed expert system rules. TARS reads constrained natural language text, reduces it to a regularized English form, and then matches the regularized statements to expert system rule templates. The knowledge in the regularized statements is then output in expert system rules, in this case, CLIPS. (Slides 13 and 14.)

We have also used this technology to convert an expert's problem solving steps elicited in AKAT to CLIPS rules. An intermediate form of the procedural knowledge (IF-THEN rules) was output by a variation of AKAT and translated by TARS. The outcome is that we can load the rule base into the CLIPS environment for use as the expert system of an ITS or use as a job-aid.

Summary

The problem we have been addressing is that knowledge exists in many forms and is sometimes needed in different formats. Our approach is to develop technologies that facilitate the translation of knowledge in order to produce computerized training or job-aiding systems. We have investigated ways of eliciting and representing undocumented expert's procedural knowledge for use in hypermedia, intelligent tutors, and indirectly expert systems. We have also explored NLP for reconstructing experts' procedural knowledge or knowledge embedded in procedural guides into a representation useable by ITS or expert systems. The goal is to have the capability to develop computerized training or job-aiding systems more efficiently. (Slide 15.)

COMPUTER-ASSISTED KNOWLEDGE ACQUISITION FOR HYPERMEDIA

AIR FORCE SYSTEMS COMMAND

DR KURT STEUCK

AF HUMAN RESOURCES LAB

4 DEC 90

OUTLINE

- BACKGROUND
- PROBLEM
- AUTOMATED KNOWLEDGE ACQUISITION TOOLS
- HYPERMEDIA
- EXPANDED APPROACH
- SUMMARY

SLIDE 2

BACKGROUND

- MISSION: RESEARCH AND DEVELOP INTELLIGENT TRAINING TECHNOLOGIES
  - INTELLIGENT TUTORING SYSTEMS
  - KNOWLEDGE ACQUISITION TOOLS
  - HYPERMEDIA
  - ARTIFICIAL NEURAL NETWORKS
  - VIRTUAL WORLDS

SLIDE 3

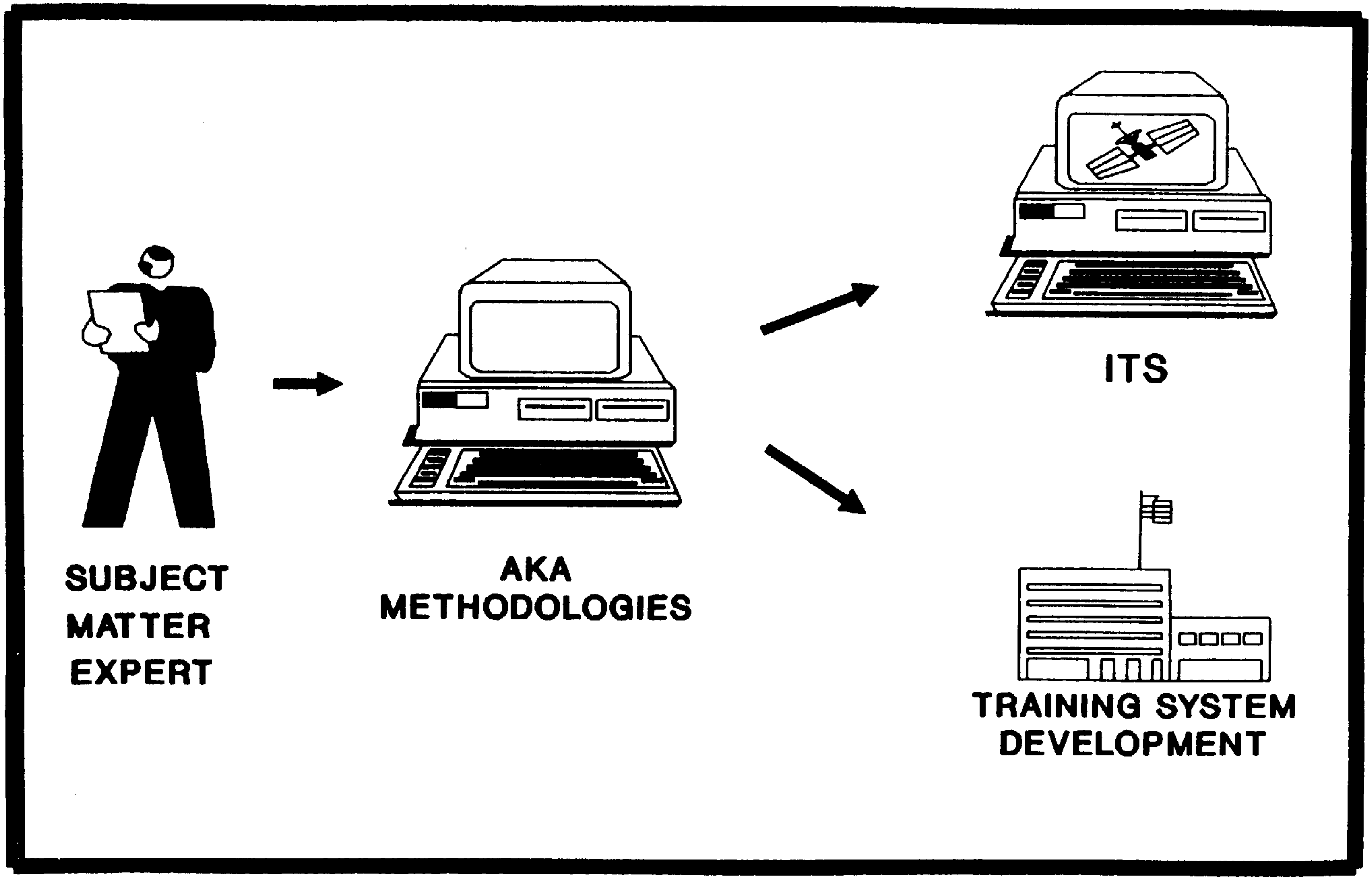
PROBLEM

- KNOWLEDGE EXISTS IN DIFFERENT FORMS
  - IN DOMAIN EXPERTS
  - IN TECHNICAL DOCUMENTS
  - DIAGRAMS, PICTURES, ETC

- KNOWLEDGE NEEDED FOR:
  - INTELLIGENT TUTORING SYSTEMS
  - HYPERMEDIA
  - TRADITIONAL COMPUTER-BASED TRAINING
  - EXPERT SYSTEMS

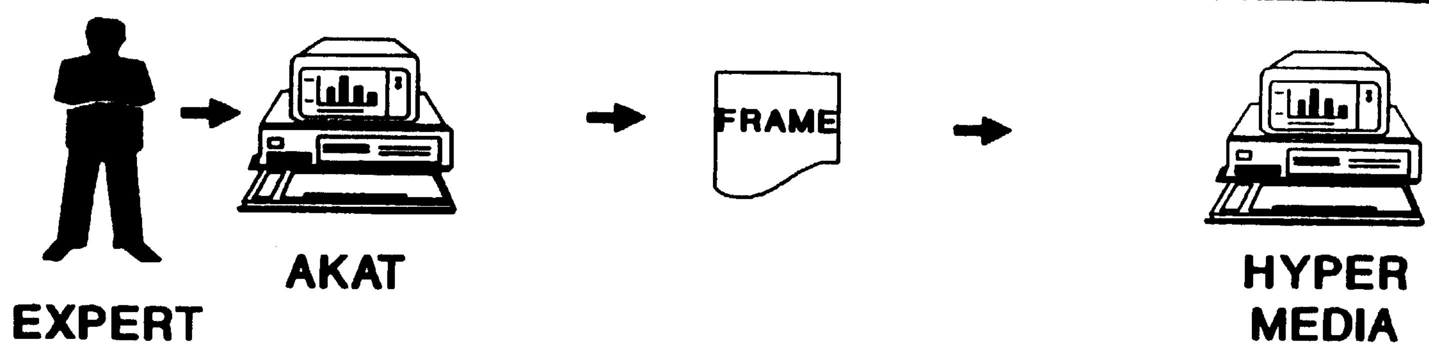
SLIDE 4

AUTOMATED KNOWLEDGE ACQUISITION



Slide 5

KNOWLEDGE ACQUISITION AND HYPERMEDIA

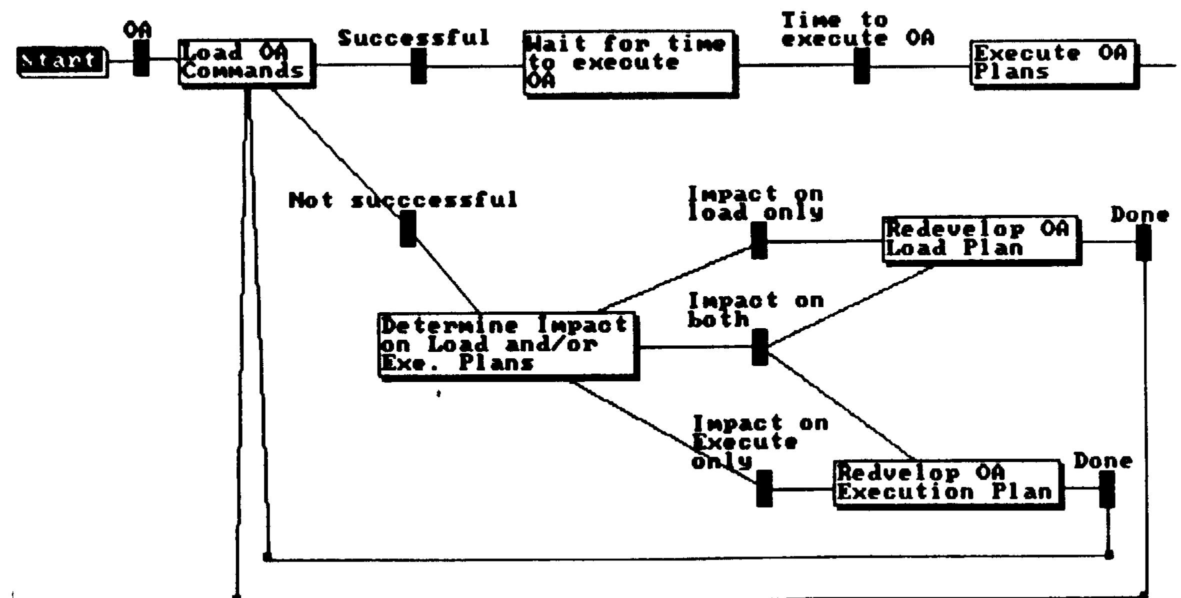


STEPS

1. KNOWLEDGE ENGINEER INTERVIEWS EXPERT
2. REPRESENT KNOWLEDGE WITH AKAT
3. OUTPUT KNOWLEDGE IN FRAME
4. READ FRAME WITH SCRIPTING LANGUAGE
5. TRAINING DEVELOPER ENTERS INSTRUCTION

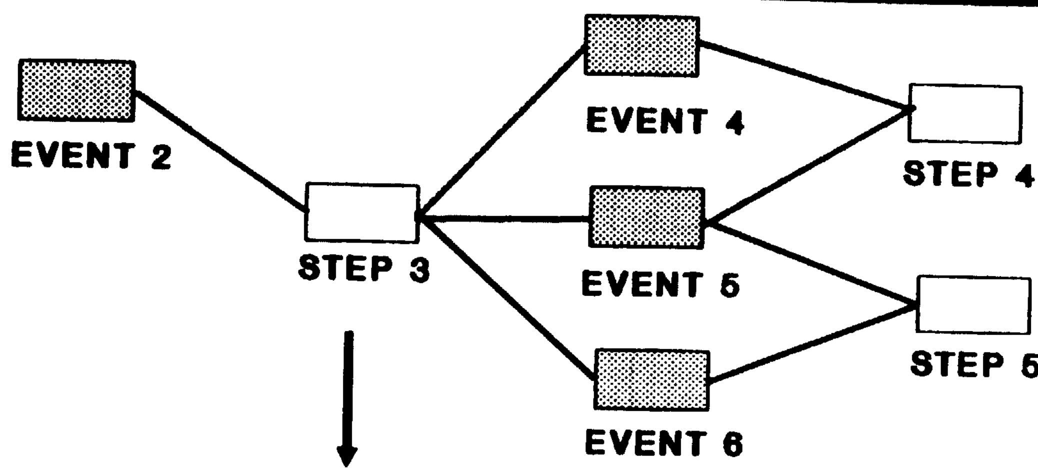
SLIDE 6

AUTOMATED KNOWLEDGE ACQUISITION



SLIDE 7

AUTOMATED KNOWLEDGE ACQUISITION



KNOWLEDGE FOR DETERMINE IMPACT ON PLANS

CONCEPTS FACTS PROCEDURES RULES	1. LOAD PLANS 2. EXECUTE PLANS

SLIDE 8

AKAT FRAME OUTPUT

STEP: DETERMINE IMPACT ON PLANS

EVENT: IMPACT ON LOAD ONLY

EVENT: IMPACT ON EXECUTE ONLY

EVENT: IMPACT ON BOTH

CONCEPT: LOAD PLANS

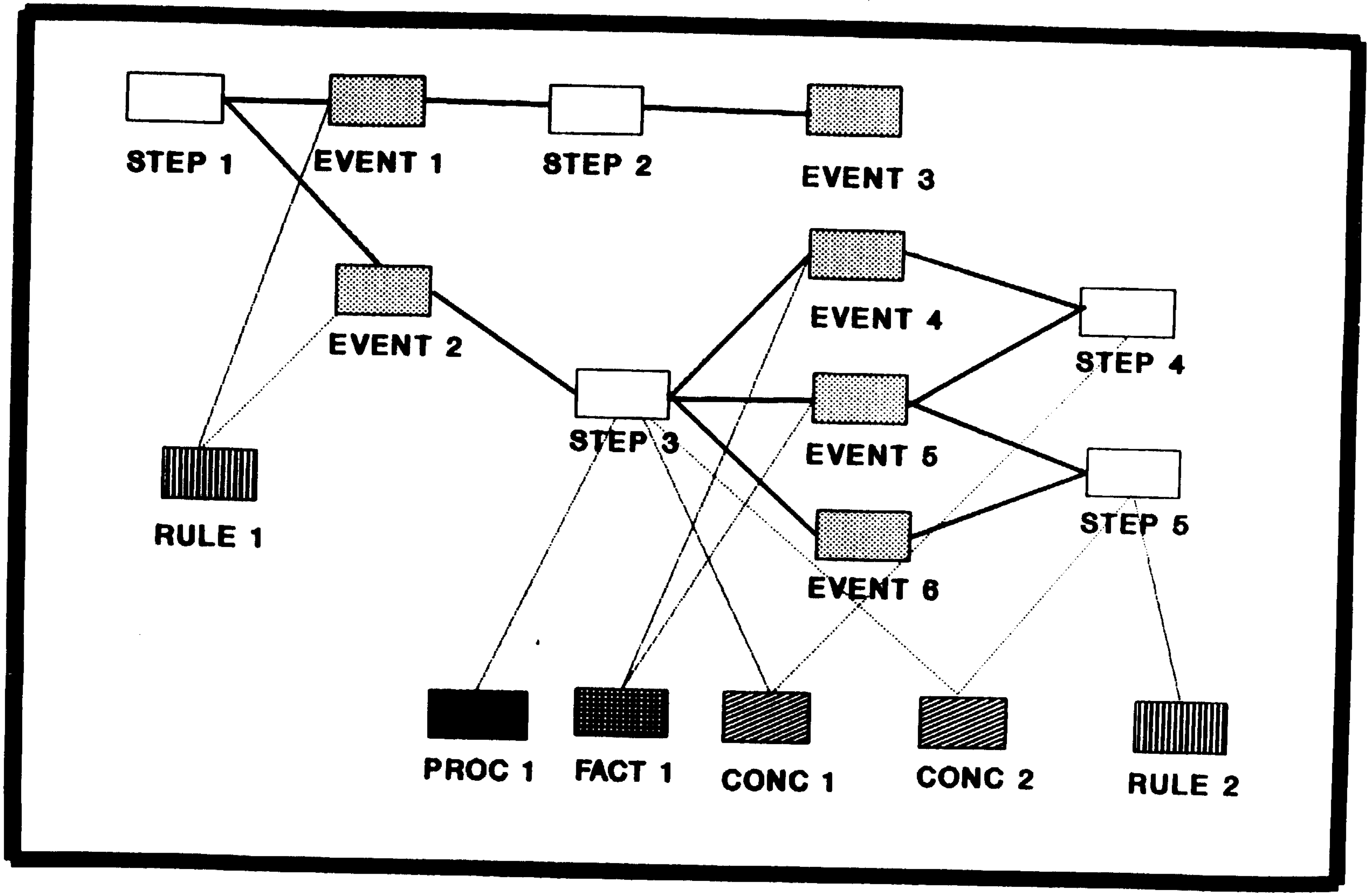
CONCEPT: EXECUTE PLANS

FACT: CHANGES ON LOAD AND EXECUTE PLANS
ARE INDEPENDENT OF EACH OTHER

PROCEDURE: COMPARE PLANNED AND ACTUAL DATA BY ...

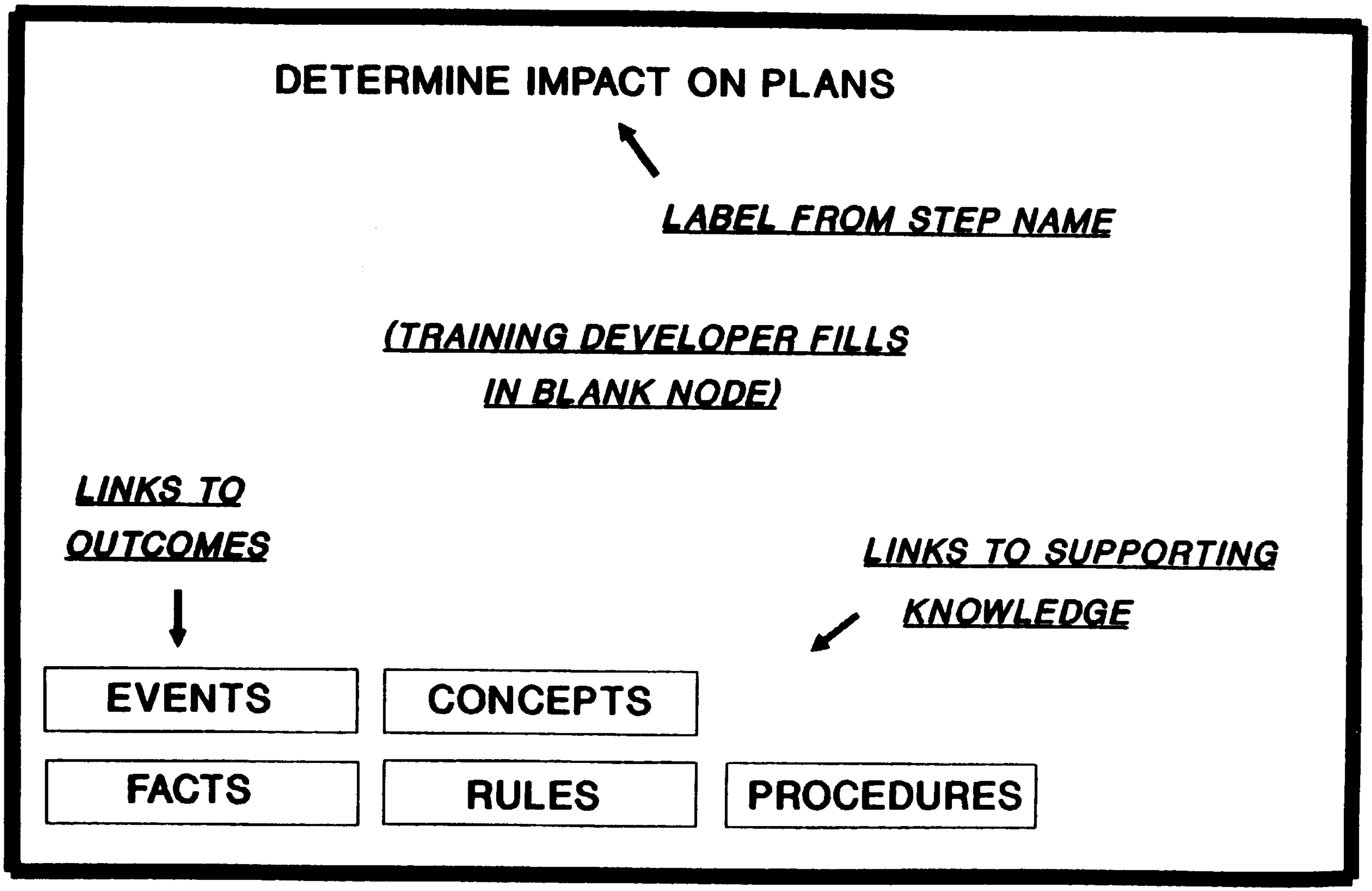
SLIDE 9

HYPERMEDIA KNOWLEDGE STRUCTURE

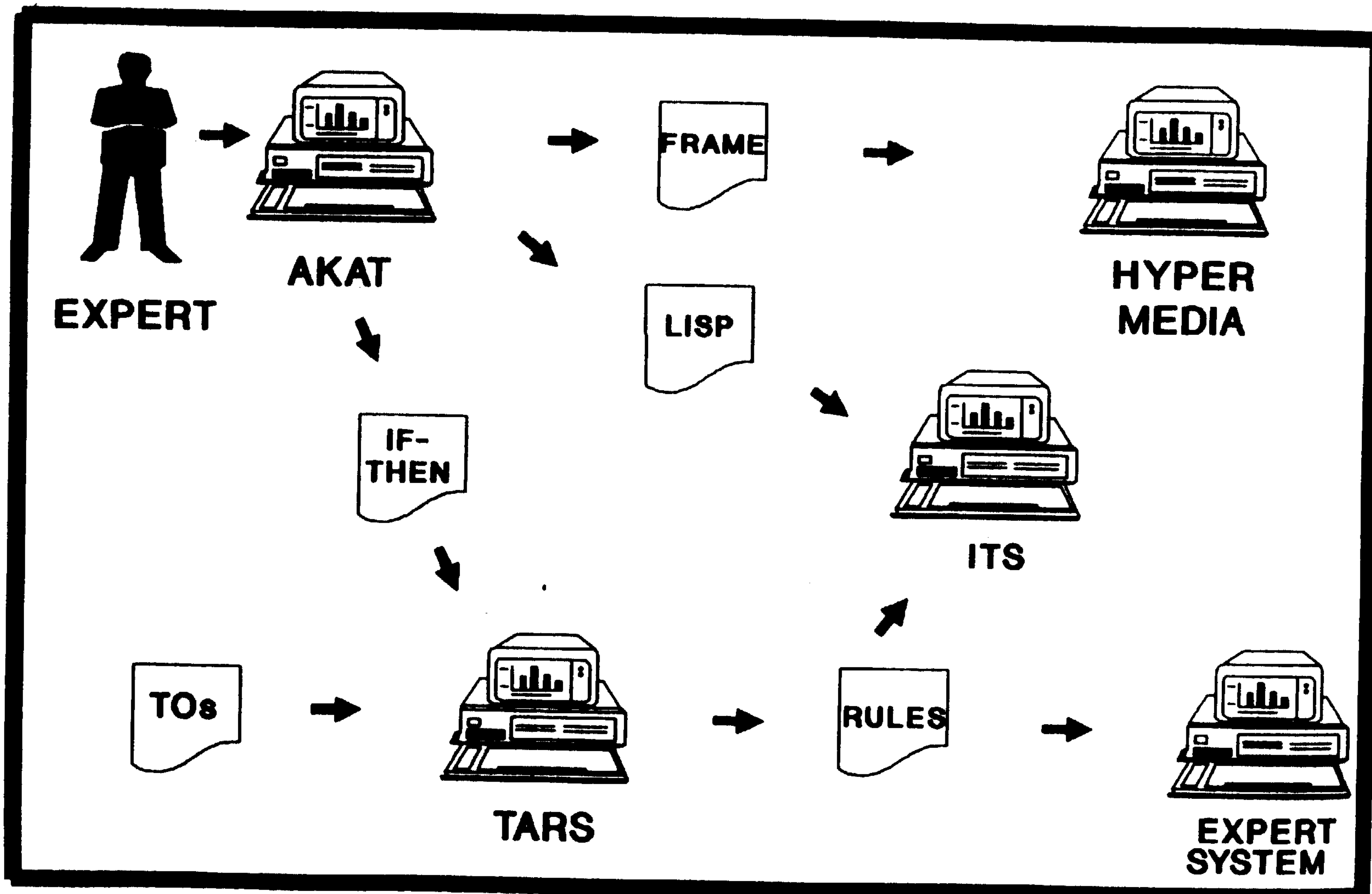


SLIDE 10

HYPERMEDIA NODE



EXPANDED APPROACH



SLIDE 12

TEXTUAL AUTOMATED REDUCTION SYSTEM (TARS)

- INPUT FROM TECHNICAL MANUALS

  "ROTATE THE AZIMUTH ADJ DIAL AS LONG AS THE TARGET INDICATOR IS ABOVE 3 VOLTS."

- INPUT FROM AKAT

  "IF LOAD OA SUCCESSFUL, THEN WAIT."

  "IF LOAD OA UNSUCCESSFUL, THEN DETERMINE IMPACT."

TEXTUAL AUTOMATED REDUCTION SYSTEM (TARS)

- OUTPUT FROM TECHNICAL MANUAL

```
(defrule rule0
(not 9<target indicator is above 3 volts> )) =>
(printout t "<ROTATE <THE AZIMUTH ADJ DIAL>> " crlf)
(printout t "IS <TARGET INDICATOR IS ABOVE 3 VOLTS>? (Y/N)"
(bind ?ans (read))
(if (eq ?ans yes) then (assert (<target indicator is
above 3 volts> ))
```

- OUTPUT FROM AKAT

```
(defrule7
(<LOAD OA UNSUCCESSFUL> ) =>
(assert (<DETERMINE IMPACT>)))
```

SLIDE 14

SUMMARY

- KNOWLEDGE
  - EXISTS IN MANY FORMS
  - MULTIPLE USES
  - NEEDED IN MANY FORMATS

- TOOLS TO HELP TRANSLATION
  - AKAT
  - HYPERMEDIA
  - TARS

SLIDE 15

N91-21945

A Model for Addressing Navigation Limitations and Metacognitive Constraints in Hypermedia Training Systems

Dr. Glenn Freedman
University of Houston - Clear Lake
December 1990

The Problem

Hypermedia training tools still suffer from limited navigational systems.

Hypermedia users are constrained by their own metacognitive limitations, schema, lack of knowledge about the developer's schema, and limitations of linking (associational) systems.

The Result

Hypermedia training systems are often difficult to use, to gain optimal value from, and can be downright confusing, unless a person already knows the system, the developer, or the content.

Sample of Software Problems

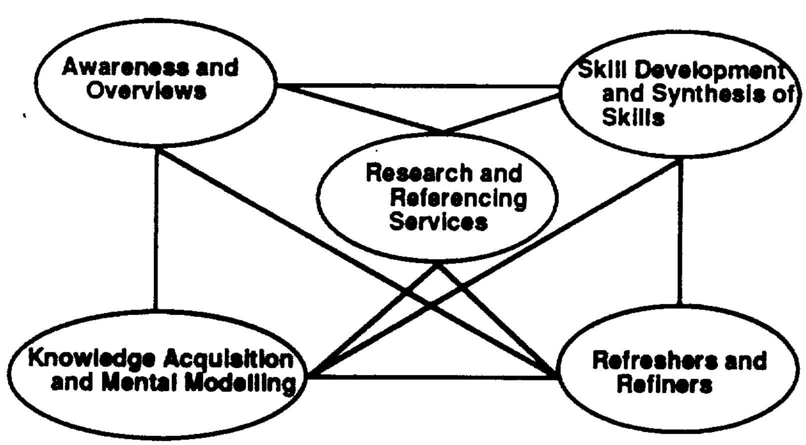
1. Difficult to browse through the system
2. Difficult to seamlessly link media types
3. Too few associational options

FILTERING, ORGANIZING, BROWSING (CONKLIN, 1987)

Sample of People Problems

1. Easily confused -- lack of cognitive 'gravity'
2. Too hard to remember everything -- can't hold one's place
3. Too many choices
4. Too much information in too many formats

Hypermedia in Training



The Three Biggest Challenges

System Design

System Design

System Design

What Are We Designing?

1. A Database System
2. A Knowledge Representation Scheme
3. A Hypertext System
4. An Instructional System
5. An Interface System
6. A Layered Layout
7. Graphics
8. An Evaluative System
9. A Search and Find System
10. A Model for Teaching
11. A Model for Learning

Strong Training System

C onfirmability

A bstraction

L ocalization

C ompleteness

I nformation hiding

U niformity

M odularity

Attributes of a Well Designed System

S afety and security

U nderstandability

P ortability

E fficiency

R eliability

M odifiability

I nteroperability

C orrectness

E xtensibility

A Design Worksheet

Instructional Goal:
Instructional Objectives:
Target Population:
Setting:
Hardware/Software Requirements:
Time Boundaries:
Proficiency Level:
Prerequisites:
Follow-up:
Node Varieties:
Link Associations:
Level of Abstraction:

Establishing a Universe of Discourse

1. Spatial Frames of Reference
2. Temporal Frames of Reference
3. Shared Domains; Coreference
4. Points of Departure
5. New Information
6. Coaching and Mentoring
7. Free Indirect Style
8. Presuppositions
9. Lexicons
10. Rules

Summary: Hypermedia Training Systems

Addressing the Metacognitive Issues:
Training for the training!

Addressing the Navigation Problems:
From Landsat to Antfarms!

Addressing the Communications Problems:
Back to the Basics!

Session 3

Issues for Real-World Hypertext Projects

Chair: Robert J. Glushko

Panel: **Three Issues for Real-World Hypertext Projects**

Robert J. Glushko
David Gunning
Bruce Warren

N91-21946

SEVEN WAYS TO MAKE A HYPERTEXT PROJECT FAIL

Robert J. Glushko
Search Technology
Norcross, GA

ABSTRACT

Hypertext is an exciting concept, but designing and developing hypertext applications of practical scale is hard. "Hypertext engineers" must overcome seven problems to make a project feasible and successful. These are (1) Developing realistic expectations in the face of hypertext hype; (2) Assembling a multidisciplinary project team; (3) Establishing and following design guidelines; (4) Dealing with installed base constraints; (5) Obtaining usable source files; (5) Finding appropriate software technology and methods; and (6) Overcoming legal uncertainties about intellectual property concerns.

INTRODUCTION

The excitement in the technical community and the popular press is rapidly inspiring others to pursue the vision of hypertext as a means to combine text, graphics, voice, and other media to make information more accessible, usable, and entertaining. Nevertheless, the novelty and immaturity of "hypertext engineering" as a discipline causes many projects to fall short of these goals. Elsewhere I have emphasized some of the challenging design issues that must be overcome "to make the hypertext vision happen" (Glushko, 1990b). In this paper I take a broader view to include management and non-technical factors to expand the list to seven ways in which hypertext projects fail. Not all of these problems are specific to hypertext projects, but together these seven issues conspire to make hypertext applications hard to design, develop, and deploy successfully.

A Composite Case Study

Attracted by the excitement about hypertext, Company C decides that links and navigation features will bring enhanced usability to a problem that has traditionally been handled in a database or document archive. Since it is the first hypertext project in Company C, all of the most talented and ambitious researchers and developers find their way onto the project. No one keeps track of how much time and effort goes into it, but after a few months a carefully hand-crafted demonstration system or prototype emerges using "Hyper-X." Hyper-X is a highly-touted program that everyone is talking and reading about as a revolutionary software advance.

The prototype system is flashy and compelling on the 19-inch workstation screen. It contains only a tiny amount of the information contained in the application it is intended to replace, but even a casual observer is impressed by how usable and appealing it looks. The colors, graphics scanned from documents and books, animation, and sound effects come together seamlessly to create a multimedia proof-of-concept.

The Vice President of Company C compares the prototype with the original text-based system and declares the Hyper-X version a smashing success. All that remains is to "scale it up" by converting the remainder of the information to hypertext.

SEVEN WAYS TO FAIL

How you interpret this case study so far might depend on where you are in your project, or whether you are part of the prototype team or a member of the team tasked to scale it up. But the stage is now set for disaster. In the sections that follow I analyze the seven ways in which the project can fail based on the composite case study.

Unrealistic Expectations About Scale and Readiness

When the full-scale development project begins, the organization tasked to carry it out usually has unrealistic expectations about how hard it is and about the capabilities and resources needed to do it. Demonstration projects often bring together the best people with ample resources and whose efforts benefit from ad hoc inter-organizational cooperation because of the novelty of hypertext. Many demonstration projects "succeed" by using methods and tools that are impossible to scale up (Alschuler, 1989). Often the demonstration project uses an off-the-shelf software package that provides neither the capacity nor the performance to deliver the bulk of information now managed by the traditional database or file system. Worse, the information examples and links in the demonstration project may have been carefully hand-crafted, an unworkable approach for a system several orders of magnitude larger. An obsessive focus on "how it looks" often drives out serious consideration of "how it has to work." If a demonstration project takes three months to convert five articles from an encyclopedia into an interactive hypermedia form, how long will it take to convert a thousand articles using the same techniques? Automatic, semiautomatic, or template-based techniques are the only realistic option for large conversion projects, but these are almost never applied during the initial prototyping effort, which usually focuses on user interface concerns.

Over time, organizations will acquire an experience base that allows them to make realistic estimates for large-scale projects that involve text and hypertext. But in the meantime it is better to be conservative.

Missing Skills on Design and Development Team

Organizations considering a hypertext project usually have ample software design and development skills available. However, hypertext projects may require (or at least benefit from) a broader mix of skills. Appropriate skills in a hypertext project team include:

* Software designers
* User interface designers
* Usability testers and potential users
* Technical writers and editors
* Indexers
* Database designers

Hypermedia projects can require still other talents, including graphic artists and people who understand how to integrate multiple media, sometimes called "art directors." For conversion projects, the participation of the author or editor of the original information can be invaluable in understanding the presentation conventions of the existing format. If neither is available, the project should allow additional time to understand the design and design rationale of the existing information. Likewise, the people who run the plant, repair the airplanes, or otherwise are expert in the application area must be involved, or the project runs the risk of building a carefully-designed user interface to the wrong information.

Few organizations or individuals have expertise in more than a few of these areas. Even if the needed mix of skills for a project organization can be assembled by matrix management or by bringing in appropriate consultants, making such an interdisciplinary team work effectively together is no small challenge.

Few Published Case Studies or Design Guidelines

Because hypertext is a relatively new design field, there are few detailed published case studies or design guidelines that designers can readily use. Published reports about hypertext are not representative, typically biased toward small-scale demonstrations or research projects. While hypertext applications of practical scale have been successfully designed and implemented, in general such projects are not documented in the literature because of resource constraints in development organizations, proprietary considerations, or because they are classified for security reasons.

While there is a growing body of empirical research evaluating particular hypertext systems or specific design options, this work does not usually generalize well. In addition, what formal experiments have been able to establish is that most of the design choices, when considered in isolation, have only small percentage impacts on system usability (Nielsen, 1989). To complicate matters, sometimes small changes in the user's task or the structure of the hypertext can lead to conflicting answers about the relative merit of design features (Wright & Lickorish, 1990). Individual differences in users, especially motivational differences, and the effects of different tasks seem to have major effects on system usability. Yet who the users are and the tasks they want to carry out are often not something the hypertext system designer can control.

However, designers of hypertext systems can take steps to ensure that their systems are acceptable and effective for their users. While empirically validated design guidelines remain some way off, design methodologies for hypertext are being proposed; the most comprehensive of these is that of Perlman (Perlman, 1989).

Installed Base Constraints

Hypertext demonstration projects are often done in research organizations that have advanced technology, including workstations and high-resolution 19-inch monitors. HyperCard on the Macintosh computer is also a very popular environment for demonstration projects.

In contrast, the users for whom full-scale versions of these demonstration systems must be targeted often work with an older or different installed base of computing equipment. This installed base may consist predominantly of IBM AT-compatible processors with small display screens having limited graphics resolution.

This situation often poses a dilemma for hypertext projects. Advanced technology may be needed to demonstrate the benefits of hypertext capabilities, but the presentation of these capabilities in the demonstration projects exceeds what the installed base will support. It is essential that the funding or marketing organization promoting the project know the costs and tradeoffs implied by various technology alternatives. Which is more successful, a project that uses less-advanced technology to create lower expectations that can be met, or a project that uses state-of-the art technology that is not readily available for the average user? There is no right answer, but it is essential to ask the question when project goals are being established.

In any case, the installed base slowly changes, so hypertext designs should be prepared to take advantage of new technology capabilities as they become available. A key consideration is separating the back end and front end of the hypertext system so that the user interface can be enhanced as the installed base permits it. Temporary constraints in processing power for installed base computers can be overcome by exploiting space vs. time tradeoffs in hypertext designs. For example, navigation maps can be pre-computed and stored on a CD-ROM instead of computed in real time.

Poor Quality or Availability of Source Files

Many hypertext conversion projects are plagued by the poor quality or availability of source files. Many documents have no digital form, and even when one exists, unless a hypertext version was planned or contracted for when the documents were created, the existing digital form may not be readily usable.

One common limitation derives from the language with which the digital version of the document was composed or marked-up. The ideal situation is when the markup language is SGML (Standard Generalized Markup Language) or some other language that specifies the logical structure of the document rather than its presentation.

Optical character recognition (OCR) technology is rapidly improving, and new OCR devices that output text in SGML form are especially promising (Grygo, 1989). Nevertheless, error rates are non-negligible, so proofreading is always required, and the nature of the residual errors in OCR documents makes manual text entry viable unless correct recognition rates exceed 98% (Cushman, Ojha, & Daniels, 1990).

Taken together, potential problems with source files make it essential that hypertext projects carefully investigate source quality and availability before committing to a project schedule. A single document sample may not be representative; often a large document or document collection (such as the complete set of manuals for a large system) was assembled from parts created by different vendors or subcontractors. Each supplier may have provided documents in a different source form. If documents are obtained in various source formats, it is generally more cost-effective to have a third-party text conversion service transform all of them to a common format than to use project software resources to carry out the conversion.

Fortunately, as the "neutral data" specifications being developed as part of CALS (Department of Defense, 1985) and IMIS (Thomas & Clay, 1988) take hold, building hypertext applications on existing information will become significantly easier.

Hypertext projects whose application involves periodic publication of text created elsewhere should define formatting standards and quality control procedures for the organization that produces the information. These measures can lead to substantial improvement in the productivity of hypertext conversion by enabling the development of automatic conversion software.

Lack of Appropriate Software Tools

Most off-the-shelf hypertext software is oriented toward creating new hypertexts and is not well-suited for converting existing documents (Alschuler, 1989; Glushko, 1990a; Glushko, 1990b). Demonstration projects often use this software to create expectations about the look and feel of a full-scale implementation, and it often comes as a harsh shock to discover fundamental limitations in the software that jeopardize the viability of a project.

It may be worth waiting for the next generation of hypertext software that directly supports conversion. The CALS initiative has prompted many vendors to enter this market, and the tools are expected to improve rapidly (Smithmidford, 1989). Alternatively, some database programs or expert system shells may better support hypertext features than programs that call themselves hypertext.

If off-the-shelf software must be used for a hypertext project, it is imperative that any demonstration or proof-of-concept phase carefully address pragmatic issues of scaling up. These include both capacity concerns -- can the program manage significantly larger amounts of information with acceptable performance -- and resource concerns -- does the program imply or impose design methods for defining units, links, or other features that are infeasible when applied on a large scale (Glushko, 1990a, 1990b)?

Legal Uncertainties

In recent years there has been a rash of "look and feel" copyright infringement lawsuits and similar claims for software patents. These legal controversies have arisen because software has been defined both as a kind of "literary work," which makes it copyrightable, and as a kind of machine or method of operating one, which makes it patentable. While these legal analogies may be wrong and may someday be corrected by a new intellectual property law that recognizes the special character of software (Samuelson, 1989b), today software designers and developers are faced with chaos, uncertainty, and legal action.

As unclear as the situation is for software in general, the novel character of hypertext and hypermedia software raises still more complexities for intellectual property law. For example, if copyright law has different rules for "literary works," "audiovisual works," "sound recordings," and "pictorial works," into what legal category does an interactive hypermedia encyclopedia or a talking book fall? Are new links or notes in a hypertext system considered "derivative works" under copyright law? These and other issues are not just legal curiosities; they will have considerable impact on the legal protection available and hence the economic viability of hypermedia systems.

One aspect of copyright infringement that confronts hypermedia designers and developers is clear and well-known, but new technology has made it easier to break the law. OCR, scanners, digital samplers and video "frame grabbers" are among the wide variety of technology that makes it possible to copy almost anything and incorporate it into a hypermedia system. But having the technology does not imply the right, and a sure way to invite a lawsuit is to assume that it does. It is not a coincidence that many hypertext applications have used government documents like standards and regulations that are free of copyright restrictions.

The best defense against a copyright infringement claim is to be able to prove independent development, so keeping careful documentation of design decisions is essential. In addition, designs based on experiments or evaluations give the design a "functional" character that narrows the scope of copyright infringement claims. It is best to follow the golden rule when designing a system: Borrow from others no more than you would have them borrow from you. An alternative formulation of this principle can be found in a well-reasoned paper that presents both sides of the look-and-feel debate: Let he who has never borrowed cast the first lawsuit (Samuelson, 1989a).

SUMMARY

Hypertext is an attractive vision, but practical hypertext applications are hard to build. Disciplined approaches to analyzing information, identifying constraints in its structure and in the task environment, and using the appropriate implementation technology are required. Successful hypertext projects are those that take a cautious approach to problems of scale and that make the right tradeoffs along the way.

REFERENCES

Alschuler, L. (1989). Hand-crafted hypertext: Lessons from the ACM experiment. In E. Barrett (Ed.), *The Society of Text: Hypertext, Hypermedia, and the Social Construction of Information* (pp. 343-361). MIT Press.

Cushman, W., Ojha, P., & Daniels, C. (1990). Usable OCR: What are the minimum performance requirements. *Proceedings of the CHI '90 Conference on Human Factors in Computing Systems*, 145-151.

Department of Defense (1985). *Computer-aided acquisition and logistic support.* Washington, DC: Office of the Secretary of Defense CALS Office.

Glushko, R.J. (1990a). Using off-the-shelf software to create a hypertext electronic encyclopedia. *Technical Communication*, 37(1), February 1990, 28-33.

Glushko, R.J. (1990b). Visions of grandeur? *Unix Review*, 8(2), 70-80.

Grygo, G. (1989). High-volume scanners aid conversion to CALS standard. *Digital Review*, 6(27), July 10, 1989, 29.

Nielsen, J. (1989). The matters that really matter for hypertext usability. *Hypertext '89 Proceedings*, 239-248.

Perlman, G. (1989). Asynchronous design/evaluation methods for hypertext technology development. *Hypertext '89 Proceedings*, ACM: New York, 61-81.

Samuelson, P. (1989a). Protecting user interfaces through copyright: The debate. *Proceedings of the ACM Conference on Computer-Human Interaction - CHI '89*, 97-103.

Samuelson, P. (1989b). Why the look and feel of software user interfaces should not be protected by copyright law. *Communications of the ACM*, 32(5), 563-572.

Smithmidford, R. (1989). Vendors focus on CALS conversions for existing paper documents. *Federal Computer Week*, 3(36), September 4, 1989, 38.

Thomas, D., & Clay, J. (1988). *Computer-based maintenance aids for technicians: Project final report.* Air Force Human Resources Laboratory Technical Report AFHRL-TR-87-44, Wright-Patterson Air Force Base, OH.

Wright, P., & Lickorish, A. (1990). An empirical comparison of two navigation systems for two hypertexts. In C. Green & R. McAleese (Eds.), *Hypertext: Theory into practice II.* Intellect Press.

N91-21947

The Object the Metaphor the Power and Evergreen

or The Eighth Way to Make a Hypermedia Project Fail

Bruce A. Warren
President
Warren-Forthought, inc

The Object the Metaphor the Power and Evergreen

or The Eighth Way to Make a Hypermedia Project Fail

ABSTRACT

In an effective organization neither the production process nor the worker's job stays the same for more than a few weeks. Neither can the hypermedia databases and learning tools for that job. This paper describes a patented software technique that is necessary and sufficient to keep hypermedia databases ***evergreen***, or current with the manufacturing technology. The technique has proved its validity in four years of use in petrochemical plants. This technique is based on three principles, 1–the database must be object structured, i.e., all components must retain ***visible individuality***, 2–the authoring process must use ***no metaphors***; the author must be seeing and experiencing the multimedia data objects as he creates. And 3–the hypermedia tools must possess power in the form of ***unlimited capacity.***

Tempting targets...

There are many tempting targets for hypermedia solutions. The technology appears to be here; products that might turn massive amounts of paper information into new computerized formats. However the sheer mass of static information distracts from the reality that a critical portion of it is always changing.

No job in an effective organization stays the same for more than a few weeks or months. Neither can the hypermedia databases and learning tools for that job. However, hypermedia information is harder to keep current (evergreen) than paper information because, until recently, no editing infrastructure existed.

Where are the hypermedia light tables, paste boards, whiteout, scotch tape, plate makers, printing presses, binders, word processors, etc? In the world of hypermedia this infrastructure must be entirely resident in software. This makes hypermedia authoring software so hard to develop that customers apply good tools like HyperCard to problems it can never complete. Like trying to mow Interstate 10 with a garden tractor; the first mile of freshly mowed median looks real good. Then you hit a few old tires and disappear into an overgrown culvert. Eager customers see the demo of the pretty green strip of grass, but not the 1500 miles of weeds up ahead.

And the companies with the very real and very large problems to solve are just learning to tell the difference between a garden tractor and a John Deere 4520 Diesel with a triple gang brush hog.

...hard hat multimedia

Warren-Forthought was founded by people with the experience to know the difference. We call it `hard hat multimedia'. Not because it makes a good marketing slogan, but because all our customers really wear hardhats.

We named our package to make it real. Our multimedia workstation is called MocKingbird®. The business plan that founded the company stressed the necessity that the tools we develop put the end-user in charge; that he be empowered to keep all his information and training Evergreen.

Neglecting the Evergreen principle is a deadly mistake, because even if the authoring system is free (like HyperCard) or the supplier is impeccable (like IBM), if you cannot make major changes to the finished product with only minutes of effort; your hypermedia application will wither away from disuse.

...change often and quickly

The competitive world forces all organizations and their business systems to change often and quickly if they are to survive. Even fast tools create multimedia products that are outdated by the time they are complete. Changes need to be made as the project proceeds and continuously as the organization evolves.

The Object the Metaphor the Power and Evergreen

or The Eighth Way to Make a Hypermedia Project Fail

Yet, with a few notable exceptions, all authoring systems and programming languages create products that are essentially unalterable in their finished state. Even minor changes open a Pandora's box of debugging, retesting, and re-validation. And consume a dispiriting amount of resources.

How many copies of 1-2-3 would Lotus have sold if it were an unchangeable generic balance sheet everyone had to use?! The true power of the personal computer lies in its ability to allow instant and infinite changes in what it does and how it delivers what it does to the user.

...Evergreen - a dichotomy

All our industrial customers accept as gospel the necessity of maintaining the equipment in their plants. The equipment must stay Evergreen or the product stops coming out the pipes. However, it is very difficult to transfer that same discipline to the mass of documentation required to run the plant right. This dichotomy evolves from the availability of a powerful substitute for good documentation - the collective memory of the folks who do the work. Lately, though, this collective memory has been making mistakes that kill people, as in the Phillips polyethylene plant explosion.

Government intervention, large numbers of old hands retiring, and good manufacturing practice are building tremendous pressure to document the plant technology in some other medium than brain cells. Everyone seems to agree that computer storage is the only hope. Paper manuals will not do the job. "No one reads them; they're always out-of-date" is the almost universal comment.

...outdated and dangerous

Whether on paper or in a computer, no matter how good a database is when released, if it cannot be maintained it will quickly become outdated and dangerous. The only practical way to maintain highly technical plant data is with the people that use the data. Only they, as a group, know when the data and the real world don't match. The technicians must have the power and permission to keep their own data correct.

Often this issue frightens the management of a plant. "...you mean those guys can change anything in here?" "What if they screw it up?" "Can I lock this disk to keep them from messing around?"

(Remember, years ago, the howls from the accountants and MIS folks when these same managers got their PC's and spreadsheets?)

But... maybe it is preferable to have a powerful and effective multimedia database with a few easily detectable errors (or even sabotage) than the 'wing and a prayer and a midnight callout' system these managers are now using. A team of technicians with a powerful and useful tool will act like a football team. They won't sabotage their teammates. (And as long as no one makes them take tests on the computer, they won't sabotage it either.)

Policy problems aside, how can it be possible for the average technician to actually maintain his own computerized information system? There are two ways to look at the answer to that question. One, teach these guys to be programmers. If you can fix multi-million dollar machinery, you probably have the mental capability to program a computer. After all it's not like these guys are entry level office workers.

..his real job

However, this is a trap that has snared many projects that attempted to 'computerize' complex documentation. We must not lose sight of the fact that, even though it can be done, becoming a programmer is not why a plant technician was hired and trained. If the computer consumes the technician's time fighting bugs and learning 'computer speak' he will not use it. His real job is to run the plant and make product. His boss never forgets that; he never forgets that.

The right answer is to seek out a database tool that becomes invisible just like any other good tool. For instance, the plant electrician does not spend hours delving into the design of his oscilloscope. He turns the knobs, uses it, and puts it back on the shelf. The mechanic doesn't get excited over his crescent wrench,

The Object the Metaphor the Power and Evergreen

or The Eighth Way to Make a Hypermedia Project Fail

he clamps it on the nut, turns it a few times and throws it back in the toolbox.

...toolness

Computer hardware and software has advanced to this state of 'toolness'. You just have to look for it and keep your computer gurus out of the way. Buy the crescent wrench, not the hacker's dream system.

The hypermedia workstation must be an integrated package, a tool, that requires no installation, or initializing, or configuration. You just plug it in, slide in a removable cartridge disk and it does all the rest. The authoring and editing tools must be packaged with interactive courses that let the technician learn the few skills he needs to be in complete control of his new tool.

Learning the new tool must take hours, not months.

The development process must be fast enough that management does not need to be persuaded to 'take a flyer' and spend a year discovering if the investment will pay off or turn into vaporware. In a month tangible useful products should appear.

...three characteristics

A computer won't become a tool in the world of a manufacturing plant floor unless it possesses three characteristics: **one-** the database must be object structured, i.e., all components must retain *visible individuality* (no stacks, no source code files, no relations, no tables, etc), **two-** the authoring process cannot use any metaphors; the author must be seeing and experiencing the multimedia data objects as he creates the hypermedia product, not some arbitrary representation of that data such as icon lists and spaghetti diagrams. And **three-** the hypermedia tools must be blessed with raw power and unlimited capacity; Lego skyscrapers are very impressive creations, but totally useless.

...visible individuality

All MocKingbird products are designed with these three principles at the core of the software design: visible individuality, no metaphors, and unlimited capacity. These three characteristics enable the content to stay Evergreen because the user can see it. Just like the real world. Anything hidden remains a mystery and is ignored. Visible individuality means the author and user can see each object and they are not combined into some stack or compound document or relational database. The Macintosh makes this easy by using the standard Finder. All MocKingbird objects (pictures, text, sounds, animations, etc) are placed in folders using the Finder. Nothing is hidden. Visible individuality also requires the elimination of the control program that contains the linkages and other hooks. MocKingbird software distributes the control commands and links to each of the electronic documents. No master program exists for the hypermedia database. Our patent, #4,877,404, covers this technique of embedding executable code in the individual graphics files, making each file a stand alone interactive object.

...no metaphor

The principle of *no metaphor* should be obvious, but never is. Hypermedia has never existed before. Only computers can do it. Contrast that with desktop publishing which is transplanted from a long tradition going back to Gutenberg. One could argue that the ultimate metaphor in hypermedia is virtual reality - what you see, hear and feel on the computer matches the world being documented on the computer. But the real world is not a metaphor, it is reality. If you use a metaphor like cards, icons, spaghetti diagrams, flow charts, etc. your authoring system will not be successful tackling a manufacturing process that is grounded in very real, very tangible equipment.

...power and capacity

Power and capacity limits are very hard to see in software. Subtle design decisions by software architects unfamiliar with the real world can severely limit a product. Software buyers learn to detect power and capacity limitations only with expensive failures. Hypermedia software must be limited only by disk space. (And that is almost unlimited now with networks and optical disk juke boxes). It must not slow down as the database gets larger or the number of users increases. It must work over networks with multiple users accessing a single copy of all critical multimedia objects. It must be fast; two or three seconds is all people are willing to wait to see results after a click. HyperCard, SuperCard and other similar products fail all these requirements. MocKingbird was designed over the past four years with these realities in mind; and delivers the power and capacity to handle any size project.

In practice it turns out the the capability to stay Evergreen hinges on the power and capacity issue. For instance, multiuser access to a single instance of a multimedia object is not only a critical power issue, but it enables that object to stay Evergreen. Multiple copies of the same data will kill a database faster than anything else because even good-faith efforts to update information will often result in missed locations that remain unchanged. Any old data is not only obsolete, but dangerous in an environment like a chemical plant.

Capacity must be unlimited or the users will quickly learn that it is too much work to condense, compress, and edit their new information to fit. Or they must throw out an object to fit another in. Anyone with a personal computer spends a lot of time going through this process with his hard disk. It is tough to decide what to throw away.

...back in the old days

Interestingly enough, back in the days of floppy disk based PC's this problem didn't exist. It was a different problem - how do I manage all these floppies so that I can find the one I need. Capacity additions were easy and limited only by filing space in the office - just buy a few more floppies. Think of a floppy disk as a storage object!

Evergreen requires that the hypermedia tool function like an old floppy based PC– the user sees no limits–while eliminating the search and retrieval problem created by thousands of floppies strewn everywhere.

The multimedia success equation can be stated this way:

Object Structured
+ No Metaphor
+ Huge Capacity
= Evergreen

Session 4

Research Activities at NASA

Chair: Robert T. Savely

Hypermedia Applications in a Mission Operations Environment

Tony J. Ames

Integrating Knowledge and Control into Hypermedia-Based Training Environments: Experiments with HyperCLIPS

Randall W. Hill, Jr.

Germ as a Tool for Space Station Documentation

Ken Crouse
Charles S. Harwick

N91-21948

HYPERMEDIA AND INTELLIGENT TUTORING APPLICATIONS IN A MISSION OPERATIONS ENVIRONMENT

Troy Ames
Code 522
Goddard Space Flight Center

and

Clifford Baker
Carlow Associates Incorporated
Fairfax, Virginia

Overview

The Automation Technology Section at Goddard Space Flight Center is investigating hypermedia, hypertext, and Intelligent Tutoring System (ITS) applications to support all phases of mission operations. Current NASA/Goddard research is addressing:

- Research into the application of hypermedia and ITS technology to improve system performance and safety in supervisory control - with an emphasis on modeling operator's intentions in the form of goals, plans, tasks, and actions.
- Review of hypermedia and ITS technology as may be applied to the tutoring of command and control languages.
- Development of a hypertext based ITS to train Flight Operations Teams (FOT) the Systems Test and Operations Language (STOL).

This presentation highlights specific hypermedia and ITS application areas, including: computer aided instruction of flight operation teams (STOL ITS) and control center software development tools (CHIMES and STOL Certification Tool)

STOL ITS

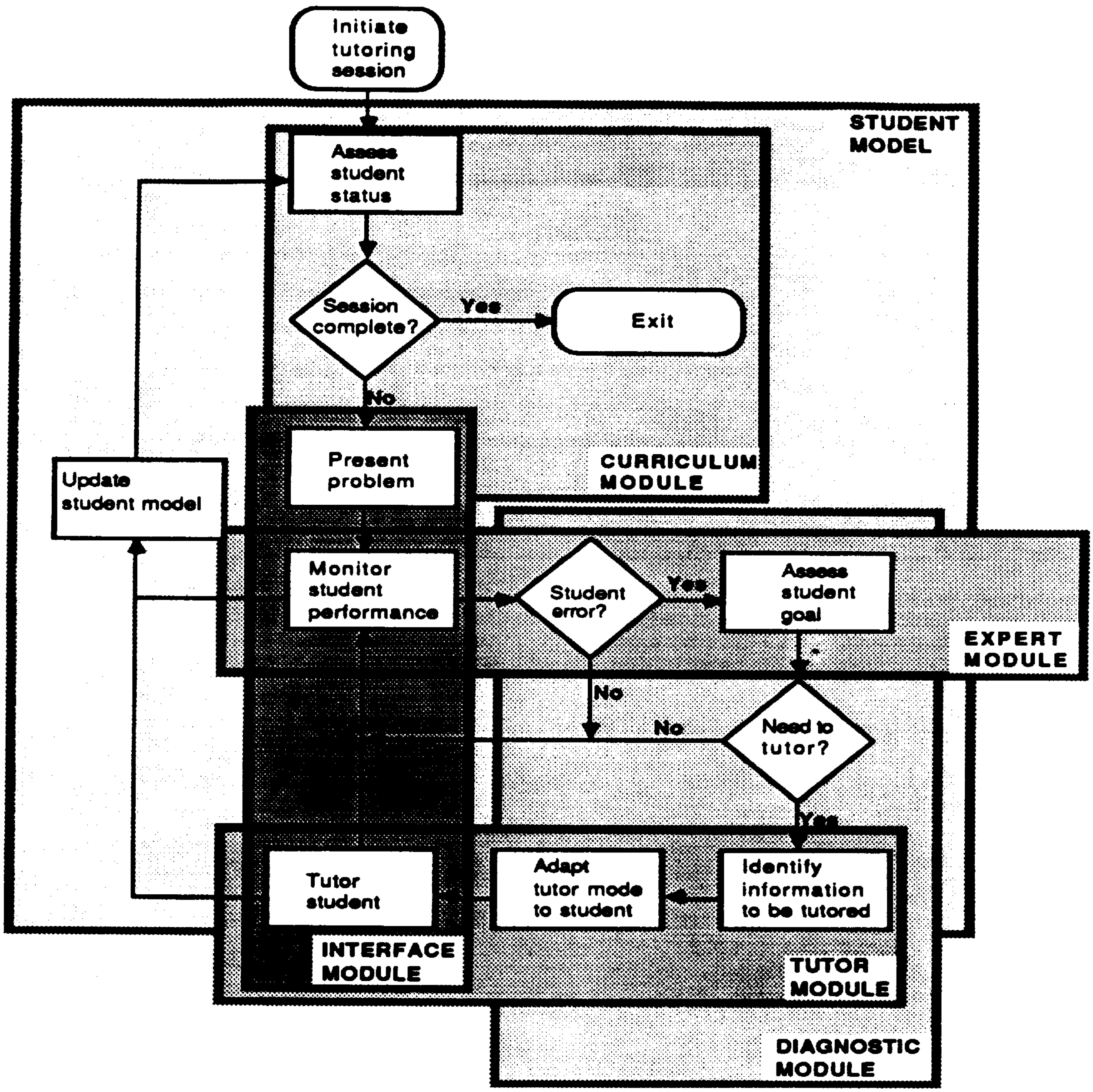
The STOL Intelligent Tutoring System (ITS) has the following design objectives:

- STOL ITS will be designed to assist NASA control center personnel in learning Systems Test and Operations Language (STOL).
- The STOL ITS will be designed to provide the Gamma Ray Observatory (GRO) Flight Operations Team (FOT) with introductory and refresher training/tutoring on STOL and its applications to the GRO/FOT.
- Develop a user interface, employing aspects of hypermedia, for an ITS to assist NASA control center personnel in learning Systems Test and Operations Language (STOL).
- Modules may serve as an ITS for other control languages such as the User Interface Language (UIL).

ITS/Hypermedia Functions

FUNCTION	FOCUS OF THIS PHASE	LEVEL OF IMPLEMENTATION Simple	Complex
Initiating the tutoring session		●	
Assessing the student's status			●
Presenting the problem	●		●
Monitoring the student's performance		●	
Assessing the student's goal		●	
Identify the information to be tutored	●		●
Adapting tutor mode to student	●		●
Tutoring the student	●		●
Updating the student model		●	

User Interface is a Central Issue in the STOL ITS Development



STOL ITS Interface Issues

Emphasize the user interface:

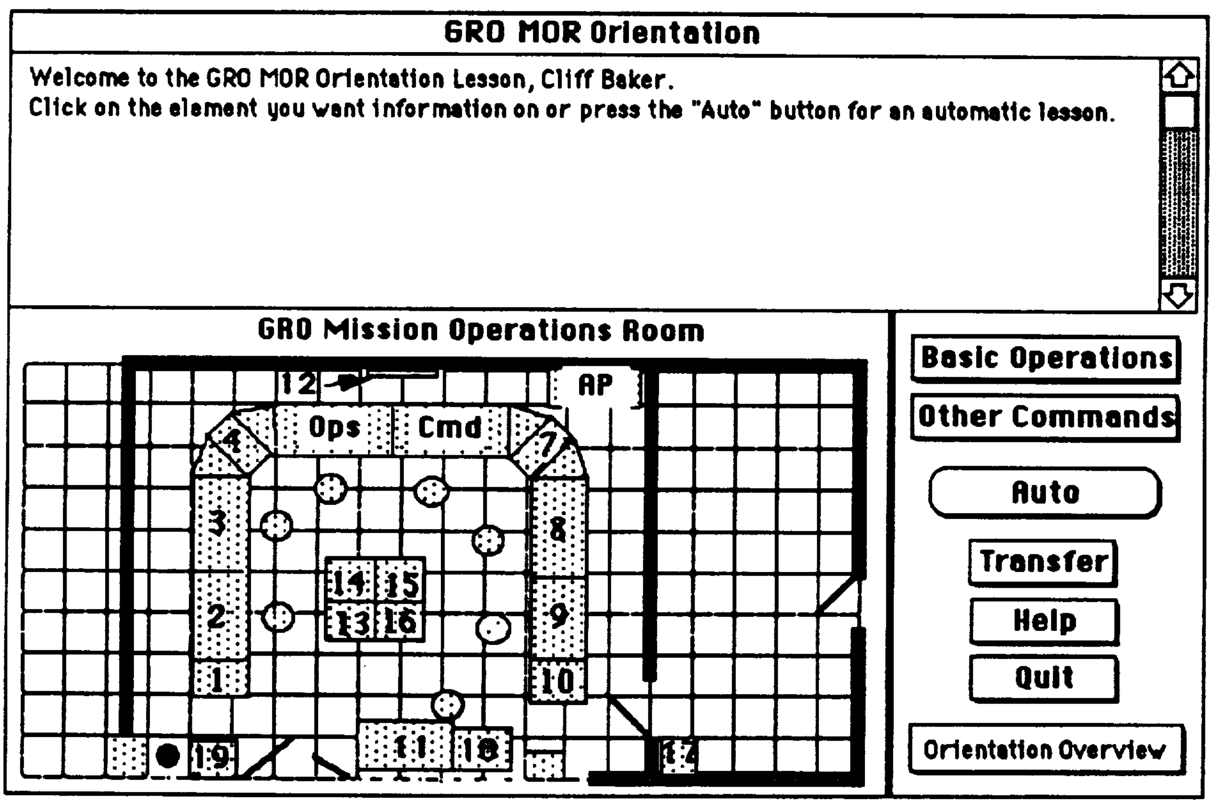
- ITS development matured to the point of becoming user-centered
- complex, relational information to be presented
- use the user interface prototype to gather user data
- use the prototype to evaluate different tutoring strategies
- use the prototype in the knowledge acquisition phase

The STOL ITS uses a hypertext interface, currently expanding towards a hypermedia environment

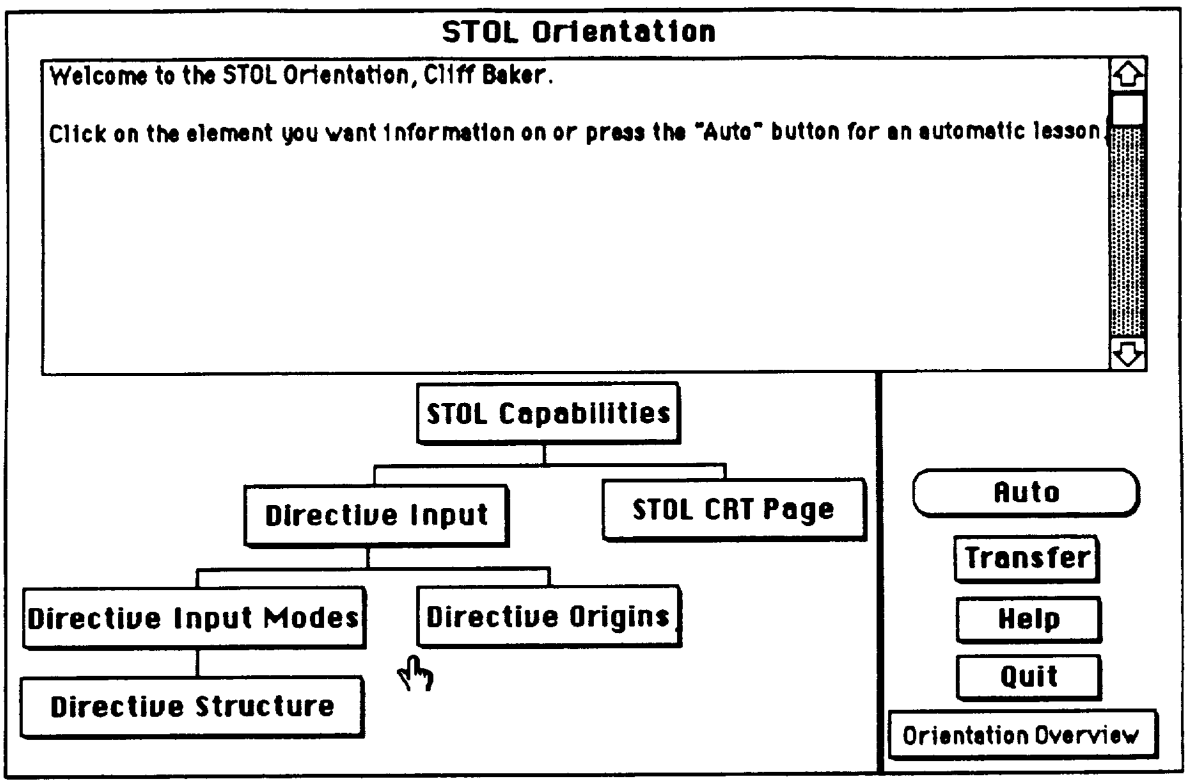
STOL User Interface

The STOL ITS user interface is:

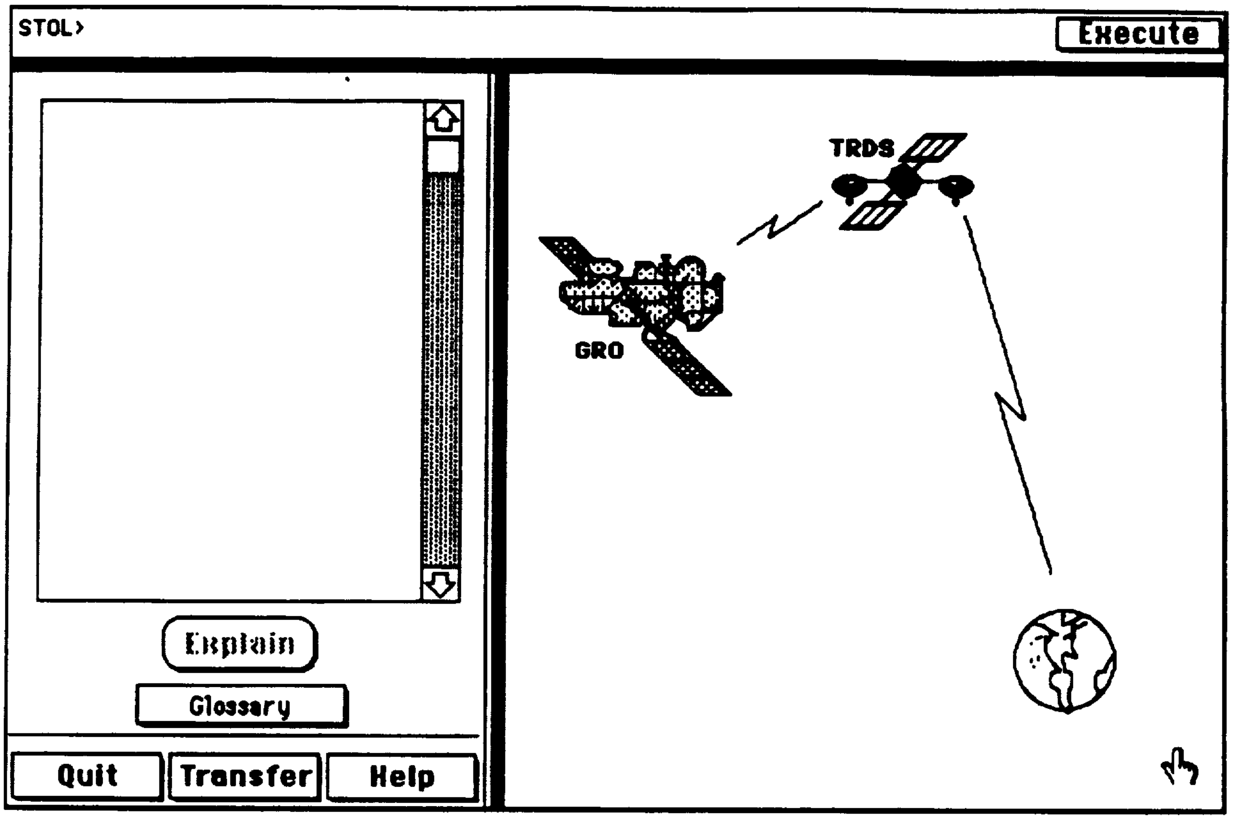
- Graphic
- Relational
- Employs animation
- Hypertext presentations
- Tending towards hypermedia



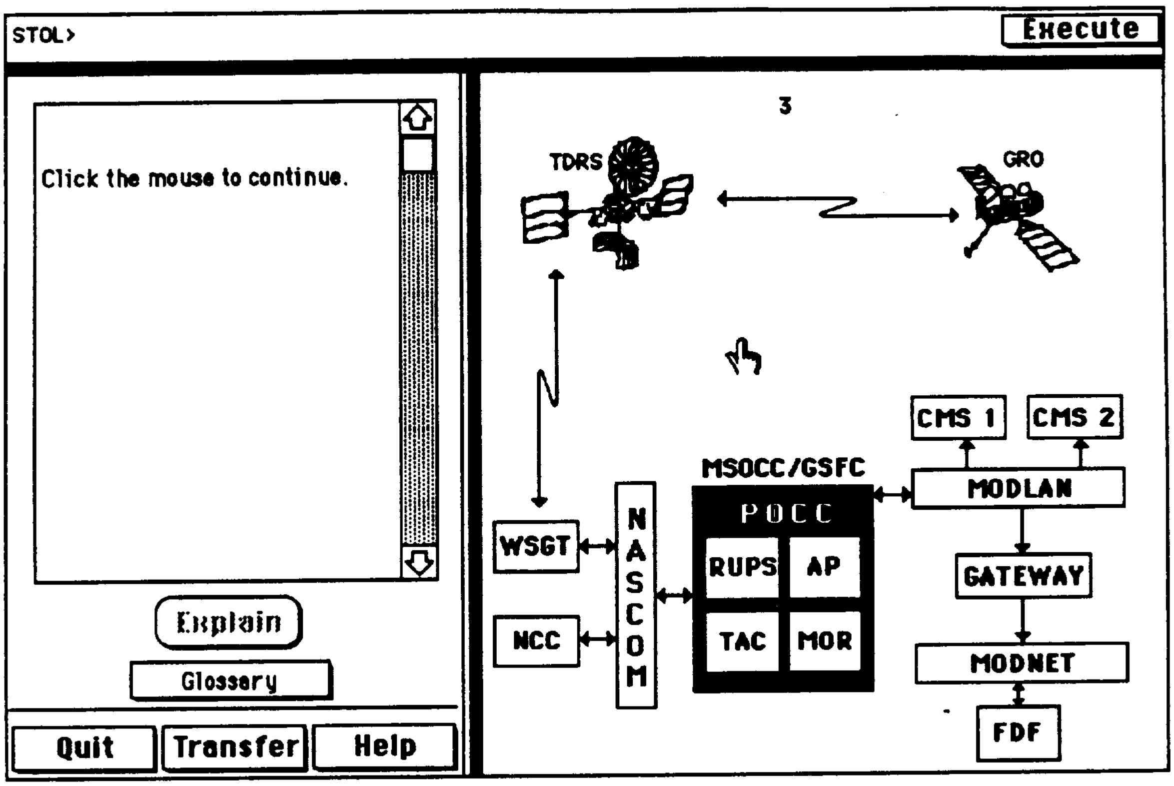
STOL ITS Orientation



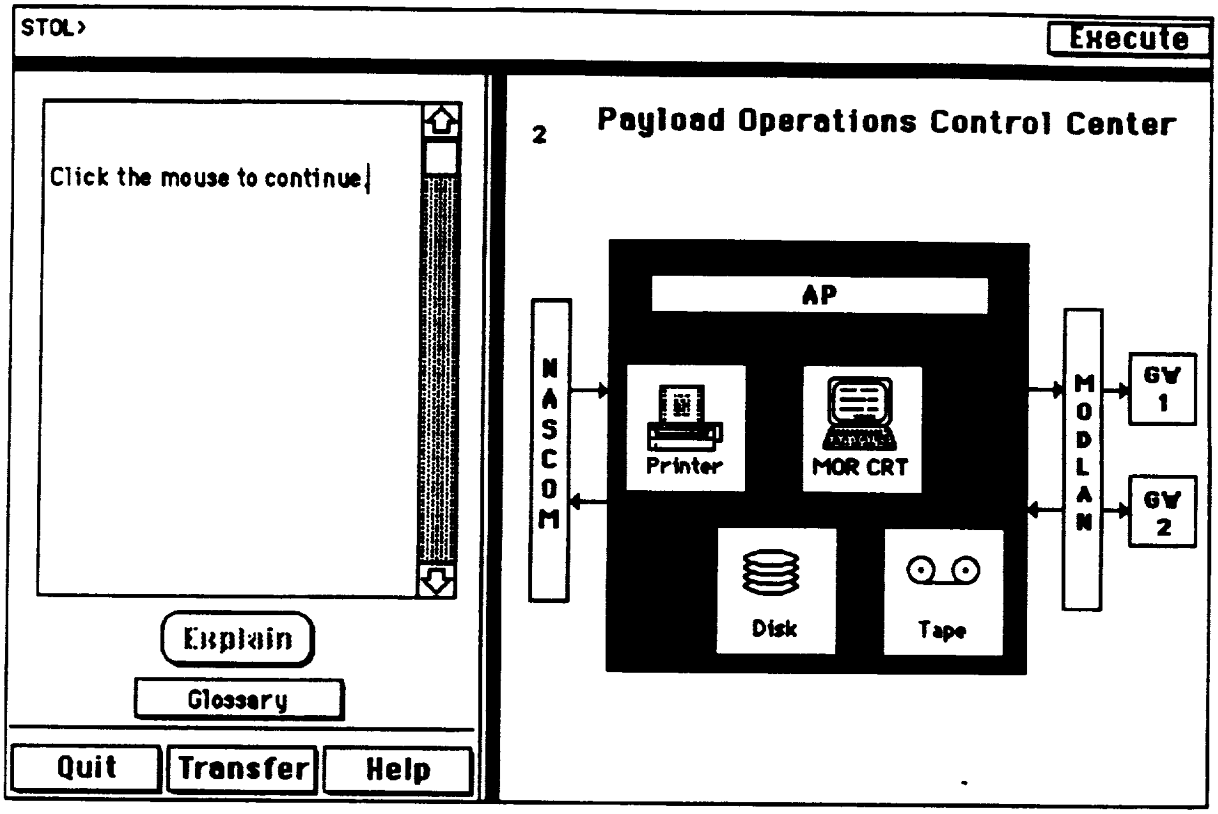
STOL Problem Presentation



Animated Feedback Provided

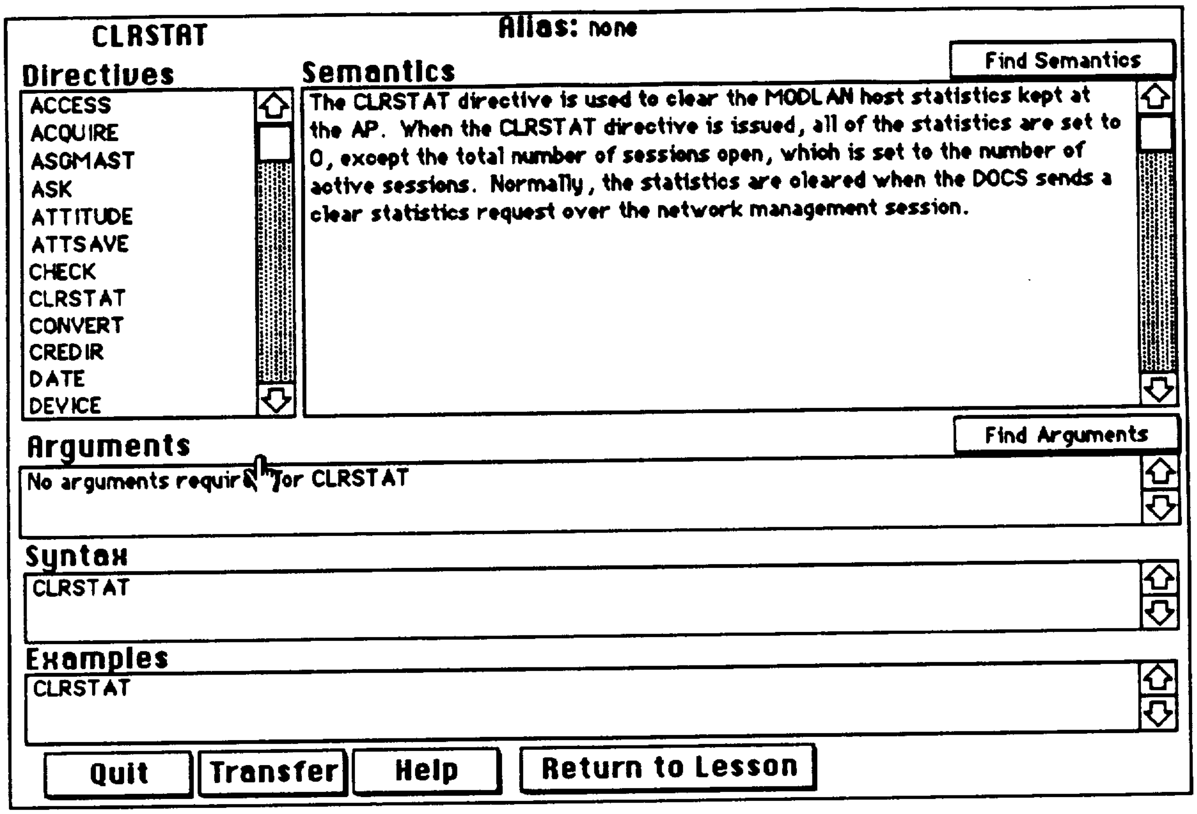


Animated Feedback (continued)



C 2

Hypertext Glossary



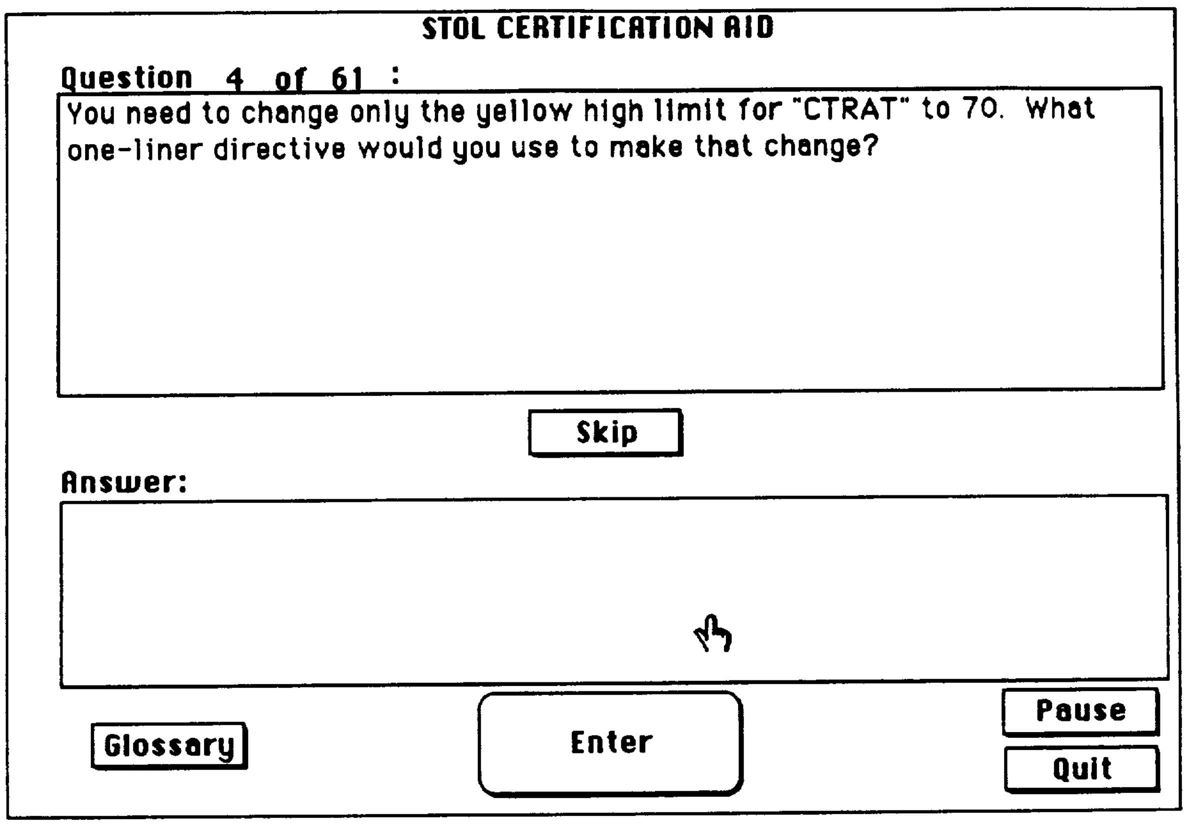
STOL Certification Tool

The STOL Certification Tool was developed to:

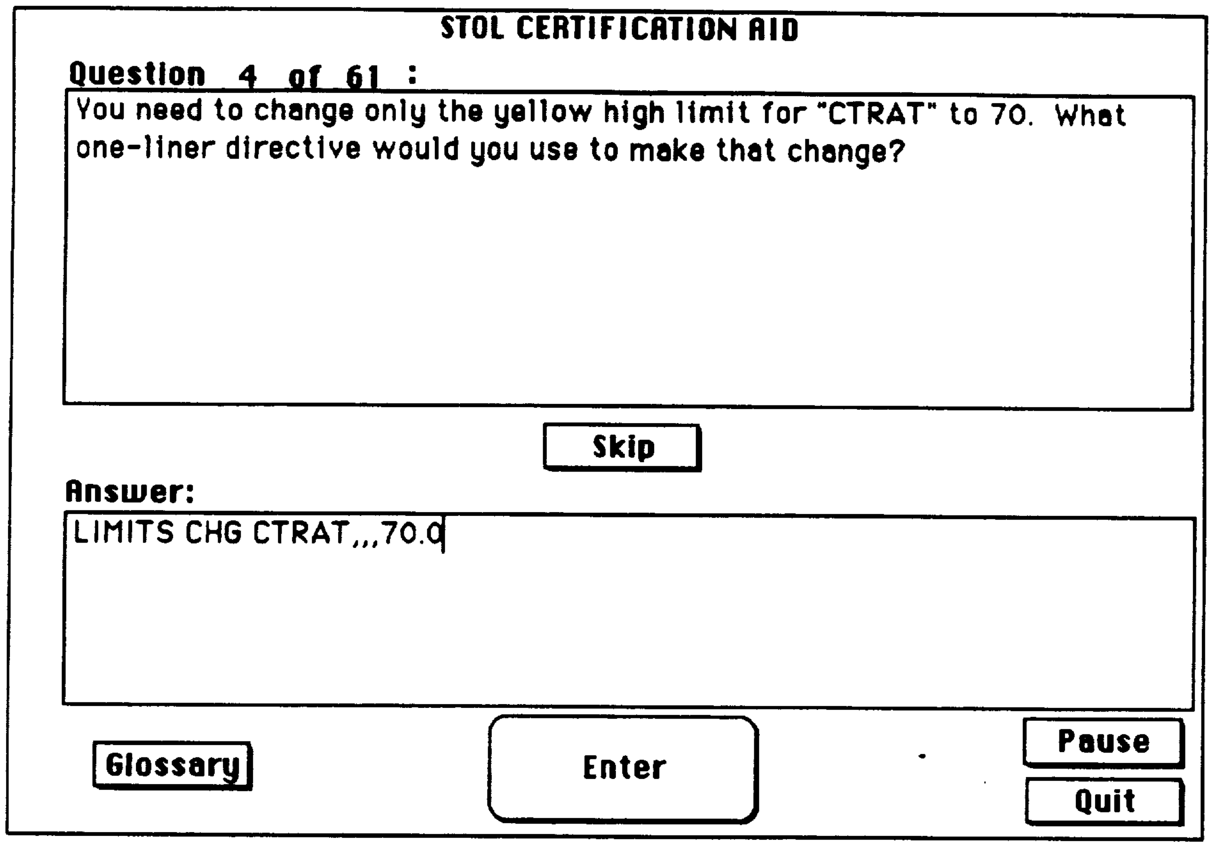
- collect error data on STOL users (commands, aliases, arguments, syntax)
- provide a basis for developing and validating STOL ITS student models
- provide a tool for certification of STOL users after and during training (including STOL ITS training)
- develop algorithms/rules for student assessments

The certification tool has a simple interface and architecture. . .

Certification Tool
Poses a Problem. . .

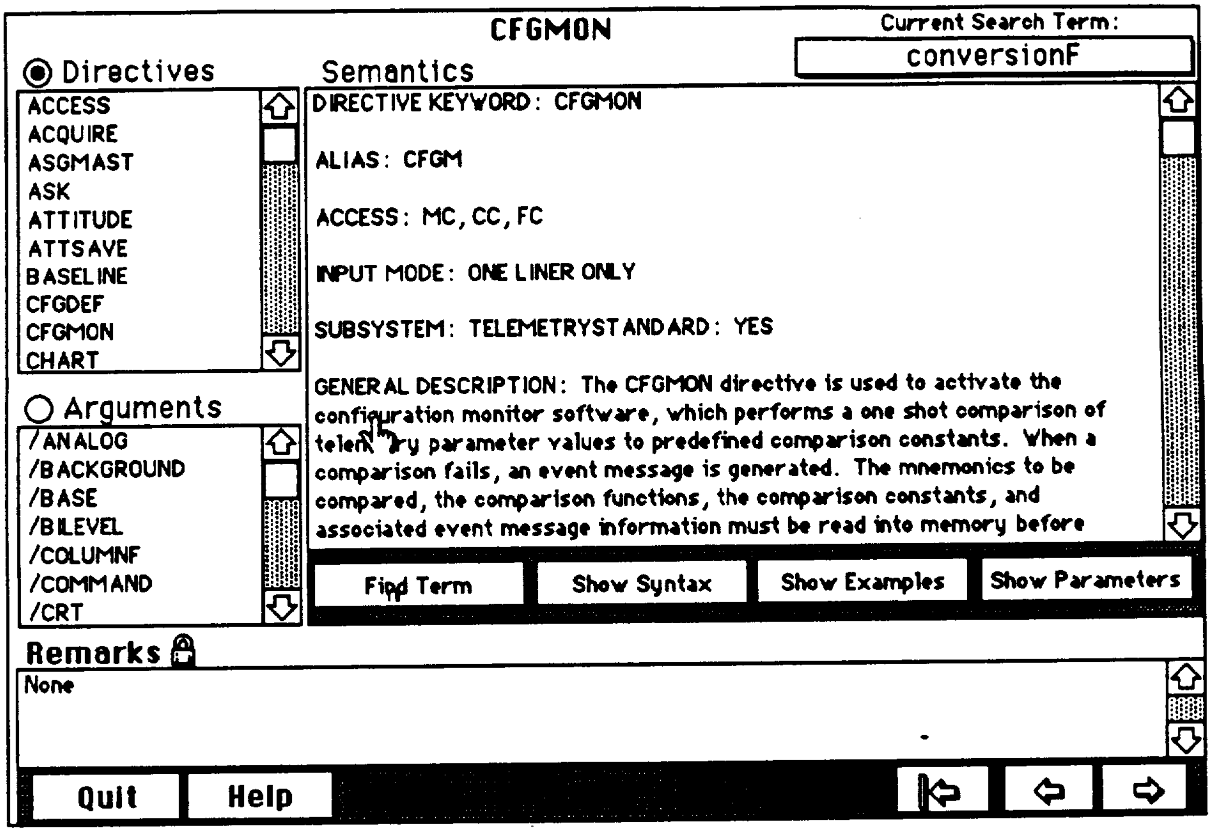


And the Student Responds. . .



After which the tool determines whether the response is correct, logs the data, and poses a subsequent problem.

The Student May Also Access a Detailed STOL Glossary



Computer-Human Interaction Models (CHIMES)

CHIMES is a prototype expert system under development which evaluates/analyzes user-computer interface designs.

CHIMES:

- accepts interface descriptions (tasks, operations)
- accepts interface designs
- initiates interface design evaluations
- summarizes interface design deficiencies
- provides recommendations for modifying/improving interfaces

CHIMES is directed at detailed interface design evaluation, addressing concerns such as:

- screen density
- visual demand
- readability
- target identification
- object manipulation

N91-21949

Integrating Knowledge and Control into Hypermedia-based Training Environments: Experiments with HyperCLIPS

Randall W. Hill, Jr.
Jet Propulsion Laboratory
4800 Oak Grove Drive
M/S 125-123
Pasadena, CA 91109

Integrating Knowledge and Control into Hypermedia-based Training Environments: Experiments with HyperCLIPS

Randall W. Hill, Jr.
Jet Propulsion Laboratory
4800 Oak Grove Drive
M/S 125-123
Pasadena, CA 91109

Abstract - In this paper we discuss the issues of knowledge representation and control in hypermedia-based training environments. Our goal is integrate the flexible presentation capability of hypermedia with a knowledge-based approach to lesson discourse management. We represent the instructional goals and their associated concepts in a knowledge representation structure called a *concept network*. Its functional usages are many: it is used to control the navigation through a presentation space, generate tests for student evaluation, and model the student, to name a few. We have implemented this architecture in HyperCLIPS, a hybrid system that creates a bridge between HyperCard®, a popular hypertext-like system used for building user interfaces to databases and other applications, and CLIPS, a highly portable government-owned expert system shell.

1. Introduction

Hypermedia-based learning environments provide two potential benefits that other computer-based instruction has typically lacked: (1) a tremendous variety of ways to present information, and (2) a relatively easy means of navigating through a non-linear information space. The advantages of being able to present instructional information in different media forms are obvious: it accommodates a wide range learning styles, it reduces boredom, and it potentially integrates information in ways that couldn't be accomplished by text alone. The second benefit, the ability to navigate through a non-linear information space, is clearly an improvement over lock-step instructional systems that are typically used in institutional training settings. This enables the student to take a self-guided tour through the hypermedia-based instructional domain without being constrained by a pre-defined control script. The problem, however, is that unconstrained exploration is not always the most efficient way to learn a task or concept. The student may never actually obtain the intended instruction due to getting lost or distracted in a confusing maze of hyper-connections. The same navigational freedom that makes hypermedia so powerful, also means there is less control over the actions of the student. How do we balance the need for instructional control with the need to explore the instructional domain?

The purpose of this paper is to discuss an approach to incorporating knowledge-based control into a hypermedia-based training environment so as to have the best of both worlds: we provide navigational freedom within a carefully selected subset of the instructional domain.To this end, the paper is organized into three sections. In section 2 we discuss the use of HyperCLIPS to manage the interface between hypermedia and a knowledge-based system. In section 3 we describe the use of the *concept network* as a knowledge representation scheme for reasoning about the instructional curriculum, the student, and the presentation. And in section 4 we describe the architecture for integrating knowledge-based control with hypermedia-based presentation.

2. HyperCLIPS = HyperCard® + CLIPS

This section of the paper discusses some practical matters concerning how we created a bridge between a knowledge-based system and a hypermedia system via a tool we have developed called HyperCLIPS. We had a strong motivation for building HyperCLIPS: we wanted to avoid having to develop either an inference engine or a hypermedia tool from scratch, hence, we sought existing tools that we could marry together via some sort of interface. Consequently, we chose HyperCard®, a hypertext system for the Apple Macintosh®, and CLIPS (C Language Integrated Production System), a highly portable government-owned expert system shell. We do not belabor the particulars of HyperCLIPS in this section: it has already been reported elsewhere [1],[2]. Rather, the purpose of this description is to highlight some of the issues involved in integrating a knowledge-based system with hypermedia, in particular, how control is exercised, and the separation of knowledge, data, and display.

2.1 Building the bridge

We built HyperCLIPS as a bridge between HyperCard and CLIPS. In theory, the interface between HyperCard and CLIPS is natural. HyperCard was designed to be extended through the use of external commands (XCMDs), and CLIPS was designed to be embedded through the use of its I/O router facilities and callable interface routines. As it turns out, there are some limitations to an XCMD: it cannot have global data and it can be no larger that 32K bytes. HyperCLIPS overcomes these limitations with an XCMD called "ClipsX", which when added to HyperCard gives it access to the CLIPS routines: *clear*, *load*, *reset*, and *run*. In addition, an I/O router was added to CLIPS to handle the communication of data between CLIPS and HyperCard.

2.2 Crossing the bridge: HyperCard to CLIPS communication

HyperCard and CLIPS do not run concurrently, rather, they pass control and data back and forth to one another. In order for HyperCard to communicate with CLIPS, it uses the *ClipsX* command along with one of four sub-commands specified as the first parameter. The four sub-commands that can be sent to CLIPS are *clear*, *load*, *reset*, and *run*. A typical use of the ClipsX command is shown in the example below, where ClipsX is called with the parameters *"run"* and *empty*.

```
-- in a HyperCard script
-- assumes the CLIPS program
-- has been loaded and is ready to run.
  ClipsX "run",empty
  get the result
  -- process the results returned from CLIPS
  get line 1 of it
  answer it with "OK"
```

HyperCLIPS Example: ClipsX with the "run" command.

Note that ClipsX is embedded as just another command in a HyperCard script, and it is given two parameters: *"run"* and *empty*. When executed, the ClipsX command deactivates HyperCard and switches over to CLIPS. The "run" parameter instructs CLIPS to execute rules that have been activated by assertions in the fact base. The *empty* parameter is sent when no data is being passed from HyperCard to CLIPS. Otherwise, a data parameter could be sent that would be trapped and parsed by CLIPS rules. Here is a summary of the four different sub-command parameters:

clear - the CLIPS command to clear the rules and facts from the CLIPS environment.
load - the CLIPS command to load a file of rules or fact definitions.
reset - the CLIPS command to initialize the rules and facts that have been loaded into the environment.
run - the CLIPS command to initiate the firing of rules that have been activated by assertions in the fact base.

2.3 Crossing the bridge: CLIPS to HyperCard communication

There are two basic commands used for passing control and data back to CLIPS:

fprintout - this is the I/O command used to pass data from CLIPS to HyperCard. The data given to the fprintout command is buffered and routed to a HyperCard variable called "the result", which is parsed and used in HyperCard.

halt - when embedded in the right hand side of a rule, the halt command suspends CLIPS and passes control back to HyperCard.

```
; in a CLIPS program
(defrule get_data
  ?f < - (phase get_data)
  =>
   (retract ?f)
   (fprintout t "need data" crlf)
   (assert (get_data_continue))
   (halt))

(defrule get_data_continue
  ?f <- (get_data_continue)
  =>
   (retract ?f)
   (bind ?data (read))
   (assert (data ?data)))
```

Example: Passing Control to HyperCard from CLIPS

In this example CLIPS sends "need data" to HyperCard and then halts so that control can be passed back also. Once CLIPS is activated again, the *get-data-continue* rule will bind the HyperCard data to a variable, which can subsequently be asserted to the fact base. Note: in addition to the *halt* command, control is also passed back to HyperCard automatically whenever the rules finish firing.

3. The Knowledge Needed to Control Instructional Interaction

3.1 The Role of Knowledge in Hypermedia

Our goal is to provide adaptive tutoring to the student by combining the flexibility of hypermedia to present the information with the representational and reasoning power of a knowledge-based system to model the student and control the interaction. We view the separation of the hypermedia from the knowledge-based system in terms of the functionality that each will implement. The hypermedia environment is used to represent information in human understandable form through the use of non-linear text, graphics, animation, video, audio, and any other available communication medium. The knowledge-based system is responsible for: (1) planning which information to present in order to meet a set of instructional goals, (2) modeling the student's apparent knowledge of the instructional domain through the use of evaluative tests and exercises, and (3) dynamically adapting the lesson to meet the needs of each student based on the modeling performed in (2). The problem, then, is linking the knowledge-based planner and student modeler with the hypermedia-based information presentation system. There must be a mapping between the information encoded in the hypermedia displays and the knowledge used to plan and monitor a lesson. The way in which we have chosen to make this link is through the use of a representation we call a *concept network*.

3.2 The Concept Network

The *concept network* is a form of a semantic net, a knowledge representation scheme that has been in use for over twenty years [3]. The concept network derives its meaning from three sources: *nodes*, *links*, and *functions* that work on the network. The *nodes* in the concept network have associations with objects in the hypermedia-based instruction environment. The nodes are divided into types, all of which have a special semantic meaning within the context of the instructional environment. They are *linked* by a set semantic relations that provide structure to the network. The *functions* take the concept network and perform operations that are keyed on the structure and meaning of the network.

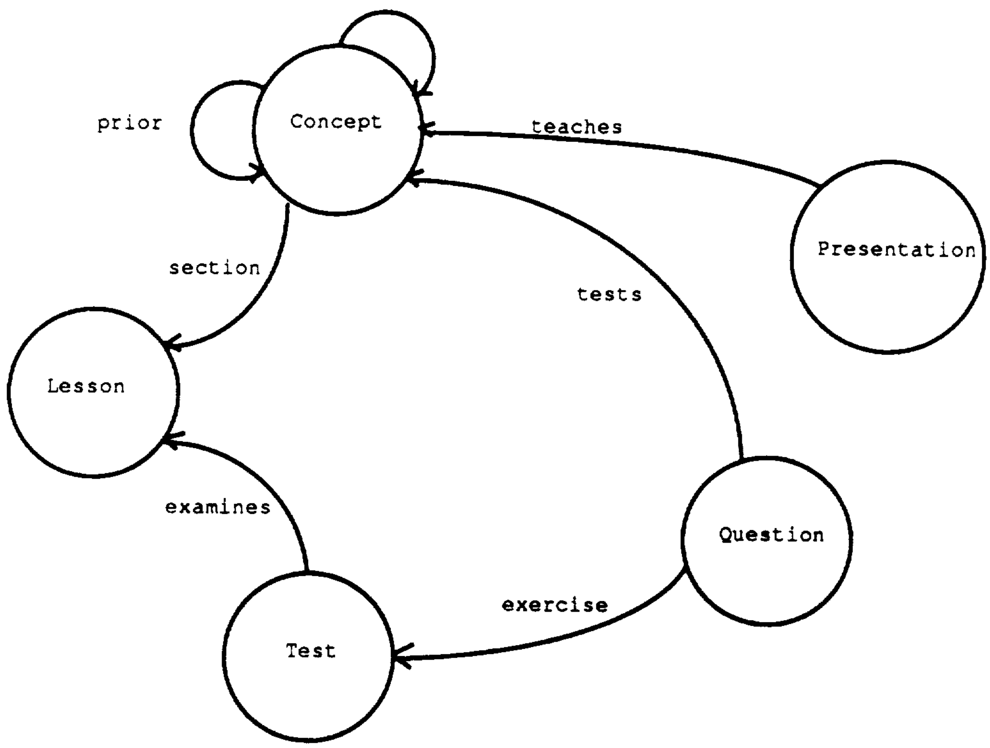


Figure 1: The Concept Network: Node and Link Types

3.2.1 Node Types

We have defined five basic node types: *concept*, *question*, *presentation*, *test*, and *lesson*. (See Figure 1 for a visual representation of the concept network.) Each node, in addition to having a name, may also have attributes associated with it to store information such as the name of a hypermedia identifier associated with the node.

Concept - This is the most commonly used node type in the concept network. A concept is a unit of knowledge or a skill; it can be decomposed into other concepts using the *subconcept* link. If a concept cannot be further decomposed (i.e. it has no subconcepts), then it is generally called a *primitive concept* and it is a self-contained unit of knowledge that must be learned by a student. Clearly, a primitive concept is a terminal (or leaf) node in the hierarchy formed by the subconcept links. Likewise, concepts that have subconcepts are internal nodes in the concept network. A non-primitive concept can be used in two different ways: (a) it can integrate several different ideas represented as concepts, or (b) it can serve as a convenient way of organizing a set of concepts together for teaching and testing.

Question - The *question* node is directly related to the concept -- a question is formulated to test the student's knowledge of a concept. It is also associated with a hypermedia object that presents the question, thus, the question node serves to link concepts with hypermedia objects. Since questions are linked to concepts, we have to carry over some of the assumptions we made about the extent of a concept (i.e. its decomposition). Questions that are associated with non-primitive concepts are assumed to test all of the subconcepts of that node. In other words, if a student answers a question associated with a non-primitive concept correctly, then it is assumed that the student knows everything about the primitive concepts that are descendents of the non-primitive concept. This is an important assumption, as we will show later in the discussion of how to analyze the student.

Presentation - A *presentation* node is similar to a question in that it serves as a link between a concept and a hypermedia object. Thus, when the instructional planner decides that a concept should be presented, it selects one of the presentations associated with the concept.

Lesson - A *lesson* is a set of concepts that must be learned. It is used to drive a lesson presentation -- or the production of a test to cover a lesson.

Test - A *test* is a set of questions that tests a lesson. A test definition node defines a way of generating a test -- the list can be composed in several different ways: a list of questions, a list of concepts, a list of lessons. Naturally, if the test is to be generated from a list of concepts or tasks, then the question selected for each concept will be from the list of questions associated with the concept.

3.2.2 Links

The nodes are connected into a network through the use of various types of links that we define below.

subconcept - The *subconcept* link denotes that one concept is a subconcept of another. In effect, the *subconcept* relation creates a hierarchy of concepts where the top level concepts are the most general and encompassing, while the terminal level concepts are the most specific and basic. This link is used a great deal in generating lesson presentation sequences as well as examinations of the material. A concept can be a subconcept of more than one concept, thus the hierarchy is not a true tree. On the other hand, we do not permit cycles in the network, making the subconcept hierarchy a directed acyclic graph.

prior - Like the subconcept link, the *prior* link is used solely among concepts. It denotes that one concept should be presented before another. Included in the reasons for using this link may be that one concept may be a prerequisite for another, or else it just makes good pedagogical sense to always present a particular concept before another. This lesson sequence generator does not require this linkage to be present, but it does take it into consideration when it is there.

teaches - Presentations are linked to concepts through the *teaches* link. This link means that a presentation, which is directly associated with a hypermedia object, contains information related to a concept. We place no limit on how many presentations can be linked to a particular concept, nor on how many concepts a presentation can teach.

tests - The *tests* link is analogous to the *teaches* link in that it links concept nodes to question nodes, which are individually associated with particular hypermedia objects. More than one question can be linked to a concept using this link. Likewise, a question can be linked to more than one concept.

examines - The *examines* link is the way we link tests with lessons. Thus, this link is a way of connecting a set of questions with a set of concepts.

exercise - Questions are linked to tests using the *exercise* link. A question can be associated with more than one test.

section - Concepts are linked to lessons via the *section* link. Note that it is not necessary to explicitly link every concept covered by a lesson to the lesson node using this link. By linking a high level concept to a lesson using *section*, all of the subconcepts of the linked node are implicitly included.

3.2.3 Functions

So far we have only described the concept network in terms of the meaning that is derived from its structure and its contents. These components are obviously important since they provide the means of semantically organizing the hypermedia to be used for instruction, but there is one more essential ingredient to the concept network that operationalizes its meaning: the *functions* that are used to transform the concept network into an actual lesson, practical exercise, or test.

The concept network nodes and links are stored declaratively as a set of assertions in a CLIPS fact base. The functions that process this collection of semantic information are implemented as CLIPS rule bases, thus, we use a form of logic programming to operationalize our concept network. To date we have implemented functions to: drive the presentation of hypermedia courseware, dynamically generate tests for a selected portion of the concept network, and evaluate the student's test results. Each of these functions are keyed to the semantic relations that exist among the concept network nodes -- much depends on the structure of the network when it comes to generating lesson sequences and tests, as will be discussed in the subsections below. In addition, functions are also influenced by the attributes of the nodes as they pertain to the student model that overlays the network. By sensitizing the presentation planner to the individual student, it can adapt the instruction to fit the needs of the individual rather than giving every student the same lesson.

3.2.3.1 The Presentation Planner - Given a concept network and a set of instructional goals (i.e. a lesson definition), the presentation planner uses chains of subconcepts to identify the set of concepts that must be taught in the lesson. The primitive concepts that must be *covered* by the lesson form a subset of the lesson concepts -- we emphasize the word *cover* because one of our basic assumptions about giving a presentation (or a test) is that in selecting hypermedia objects to present the concept, we must form a *covering set*, that is, a presentation sequence is acceptable if and only if it: (1) directly presents a primitive concept, or (2) it presents an ancestor of the primitive concept. Thus for any given primitive concept there is at least one chain of ancestors from which a presentation can be chosen that will cover it. For each primitive concept we randomly select a hypermedia presentation from the concept's ancestral chain to use for instruction, avoiding the choice of a hypermedia object that has already been chosen for another concept. Once the set of presentations has been chosen, it is necessary to order them into a sequence. The choice of a sequence is influenced by the *prior* link, which creates a partial order among individual presentations -- the rest are randomly ordered.

3.2.3.2 The Test Generator - The test generator works on many of the same assumptions as the courseware driver. Given a concept network and one or more nodes that serve as "roots" to sections of the concept network (which, you may recall, is a direct acyclic graph), the test generator produces a set of questions that will form a *covering set* for the concept subset indicated by these "roots". Note that in the instance where we want to generate a test that covers a particular lesson, we can work from our concept network definition of the *test* node to produce the test. In this case the test is linked to the lesson by the *examines* link, and the lesson has a set of concept "roots" associated with it by the *section* link.

3.2.3.3 The Student Analyzer - A student model is a representation of the current state of the student's knowledge [4]. It is used for the purpose of identifying missing knowledge as well as for diagnosing misconceptions. In the current version of our system, we concentrate on identifying missing knowledge rather than diagnosing misconceptions. Consequently, our approach to student modeling is to use the overlay method with differential analysis [5]. We assume that the concept network represents everything we want the student to know about a topic, hence it is the target model. The student's knowledge is evaluated using tests generated from the target model, and the results are overlaid on the target model. Thus, the function of the student analyzer is to take the test results and generate a model of the student. It isn't quite as easy as just marking the nodes in a concept network as "known" or "unknown" since the test does not explicitly test every concept. In the cases where a concept has not been explicitly tested, we have to infer whether the student knows the concept based on the actual results. We make these inferences by inferring whether each of the primitive concepts is known and then propagating these values backwards from the subconcepts to their superconcepts.

4. An Architecture for Hypermedia-based Instruction

In the last section we defined the structure of the knowledge used to make decisions about how to sequence a lesson, how to generate a test, and how to analyze a student. In this section we will take a step back and look at how the individual components of the knowledge-based system interact with each other and with the hypermedia environment. We will bring the pieces together to show the overall architecture of a hypermedia-based instruction system.

4.1 Preliminaries to Instruction: Analysis and Development

In order to conduct a lesson there are several prerequisites that we are assuming are in place. First, we assume that the concept network for the lesson has already been built by a course developer. Though not described in this paper, we have built a tool to assist in the construction of the network -- it includes a network editor and display capability, and it produces as output the appropriate CLIPS assertions that can be interpreted by the CLIPS rules. Note that this step may be one of the most critical since it involves a careful analysis of the structure of the knowledge to be transferred to the student during the lesson. Second, we assume that once the concept network has been built that the course developer will create a variety of presentations for the concepts in the lesson. If the concepts can be represented in a variety of alternative media forms, then the greater number of choices available to the presentation planner make for a more robust instructional environment. Third, along with the presentations being developed for the concept network, a wide variety of test questions must also be written. It is desirable for each concept to have multiple questions from which to choose when generating a test.

4.2 The Role of Knowledge in Exercising Control

So how does it all fit together? This is an important question since the issue is how much freedom should be given to the student in the presentation of the lesson. At one extreme the presentation sequence being sent to the hypermedia environment (refer to Figure 2) could be equivalent to an entire lesson, and at the other extreme it might contain the identifier for one presentation of one concept. In either case, the question of what the Hypermedia environment should do with the presentation sequence is open. One approach would be to give the student "free play" within the bounds specified by the presentation sequence, with the constraint that partial orderings among presentations be observed (i.e. don't allow free play in a region until the predecessors have been successfully traversed.) Another approach would be to present the information in strict accordance with the presentation sequence -- this approach may be appropriate in some cases, but it obviously takes away more freedom for exploration than the first approach.

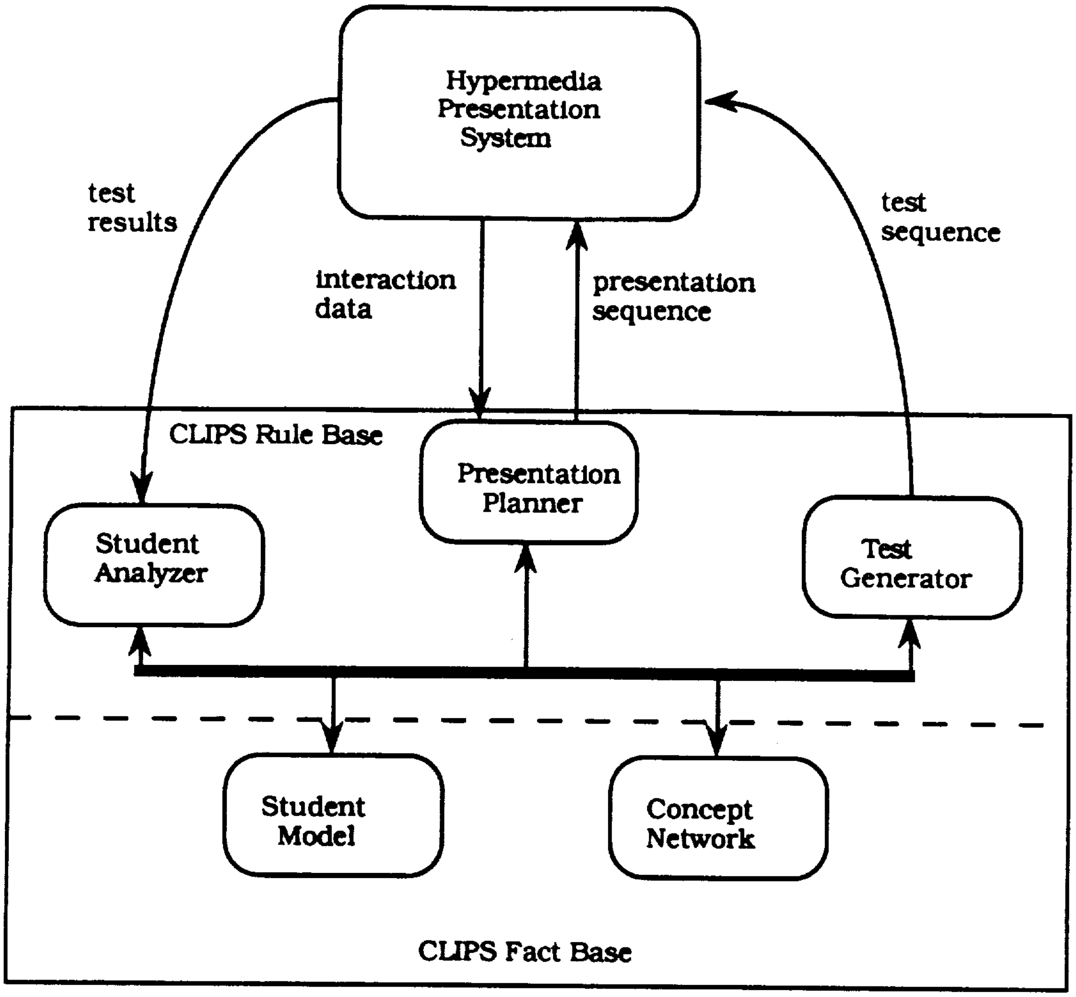


Figure 2: An Architecture for Hypermedia-based Instruction

However one decides to implement the frequency of how often control is passed back and forth between the hypermedia environment and the knowledge-based system, the general flow of control is envisioned as being a cycle:

1) generate a presentation sequence and send it to the hypermedia environment,

2) the hypermedia environment presents the lesson segment and sends user interactions back to the presentation planner,

3) the presentation planner either repeats 1, or it passes control to the test generator,

4) the test generator creates a test to cover the lesson segment just presented and sends it to the hypermedia environment,

5) the hypermedia gives the test specified by the test generator and passes the results back to the student analyzer,

6) the student analyzer takes the test results and updates the student model; it then passes control back to the presentation driver,

7) repeat the cycle until the lesson is complete or the student quits.

4.3 Current Status of Our Work

We have implemented the architecture just described using HyperCLIPS. Due to the modularity of the knowledge-based system we have implemented in CLIPS, we are confident that we could use our current knowledge-based system as a driver in other hypermedia environments. In fact, our HyperCLIPS implementation serves as a prototype and debugging tool so that we can port our efforts to a hypermedia-based instruction system called Tools for Courseware Development, which is being developed elsewhere.

5. Conclusions

In this paper we have discussed an approach to integrating hypermedia with knowledge-based systems for the purpose of conducting instruction. Hypermedia is an attractive way to give information-seekers a way of navigating through a database. It provides freedom and variety, both of which seem to be desirable traits for an information processing system. The problem with hypermedia in general is the ease of taking the freedom to navigate too far -- the user gets lost, off track, distracted, etc. These problems can be even more severe in a training system where the intent is for the user to learn some structured knowledge, be it a task or a related set of concepts. On the other hand, many computer-based instruction systems have been notoriously inflexible and boring, and could benefit from the flexibility and the variety of ways of presenting information found in hypermedia. Thus, we have suggested that by using a knowledge-based system to make decisions and exercise control, one can harness the power of the hypermedia environment without destroying its usefulness.

6. Bibliography

[1] Hill, Randall W., Jr., and William B. Pickering, "Intelligent Tutoring Using HyperCLIPS", CLIPS User Conference, 1990, Houston, Texas.

[2] Pickering, William B. and Randall W. Hill, Jr.. "HyperCLIPS: A HyperCard Interface to CLIPS". CLIPS User Conference, 1990, Houston, Texas.

[3] Wenger, Etienne, "Artificial Intelligence and Tutoring Systems", Morgan Kaufmann Publishers, Los Altos, CA, 1987.

[4] VanLehn, Kurt. "Student Modeling". In Martha C. Polson and J. Jeffrey Richardson (Eds.), Foundations of Intelligent Tutoring Systems (1988): 55-78. Hillsdale, New Jersey: Lawrence Erlbaum Associates Publishers.

[5] Wilkins, David C., William J. Clancey and Bruce G. Buchanan. "Using and Evaluating Differential Modeling in Intelligent Tutoring and Apprentice Learning Systems". In Joseph Psotka, L. Dan Massey and Sharon A. Mutter, Intelligent Tutoring Systems: Lessons Learned (1988):257-275. Hillsdale, New Jersey: Lawrence Erlbaum Associates Publishers.

N91-21950

GERM as a Tool For Space Station Documentation

by

Ken Crouse

NASA/Johnson Space Center

Charles Hardwick

University of Houston-Clear Lake

Introduction
Problem Statement

- The volume and complexity of Space Station documentation
  - Multiple levels of management
  - Division of labor into work package structure
  - Predominance of paper documentation
- Limitations of current technology

Hypermedia as a Tool for Documentation
"Why we considered hypermedia."

- Variety of types of documents
- Critical information contained in relationships between documents
- Sequential representation inadequate

Technical Approach

- Defining the problem scope
  - OMA Documents
  - RID data base
  - Relationships between documents
- Choice of tools
  - GERM - Hypermedia
  - Frame Maker - Desktop Publishing
  - Oracle - Relational DBMS

GERM

- What is GERM?
  - Developed at MCC STP
  - MCC/RICIS/JSC Cooperative Agreement
  - Prototype using proprietary software
  - Runs on Sun
- Unique Characteristics of GERM
  - Graphical interface
  - User definable schema structure
  - Links to other applications

Applications Development

- Schema file
- Icons
- Folios
- Frame Maker
- Oracle database
- Plug-in-Modules

Results

- Presentation of GERM interface structure
- User inter-action

Benefits

- Access to documents in a variety of forms
- Visual presentation of important relationships
- Management of complexity
  - Non-sequential links
  - View different levels of detail
  - Use of visual cues (color, icons)

Lessons Learned

- Need a tool that is flexible
  - Tailor graphics to applications
  - Represent different types of relationships
- Limitations
  - Does not do initial capture of information
  - Represents, but does not discover relationships

Lessons Learned
(cont'd)

- "Hooks" need to be in documents to establish relationships
- GERM is flexible enough to be used with a variety of applications beyond Space Station documentation

Conclusions

The hypermedia capabilities of GERM offer significant potential for increasing the usability of Space Station documentation.

The technology also provides capability important for design knowledge capture.

Session 5

Interfaces for Hypermedia Systems

Chair: Dona Erb

Hypertext as a Model for the Representation of Computer Languages

Randal Davis

Automating Hypertext in a Decision Support System

Michael Bieber

TEJAS: Hypermedia for the NASA Masses

Michael L. Drews

N91-21957

Automating Hypertext for Decision Support

Michael Bieber
Boston College, Carroll School of Management
Chestnut Hill, MA 02167-3808
(617) 552-3964 Email: bieberm@bcvms.bitnet

Abstract
We are constructing a decision support system (DSS) shell that can support applications in a variety of fields, e.g., engineering, manufacturing, finance. The shell provides a hypertext-style interface for *navigating* among DSS application models, data and reports. To do so we had to enhance the traditional notion of hypertext. Hypertext normally requires manually, pre-defined links. A DSS shell, however, requires that hypertext connections be built "on the fly". In this paper we discuss the role of hypertext in augmenting DSS applications and the decision-making process, and how we automatically generate hypertext nodes, links and link markers tailored to an arbitrary DSS application.

One of our main research goals is to make it easier and cheaper to build and use decision support systems (DSS). We want to enable application builders to construct DSS applications with more functionality than traditional DSSs have, and to do so more quickly. This goal led to the idea of a DSS shell supporting multiple DSS applications[1] (see Figure 1) that produce *interactive documents*, such as those in Figures 2a and 2b. Interactive documents give direct access to reports, operations, and other components of DSS applications. The navigation and clarity of presentation required by interactive documents suggest a hypertext-style interaction, but most current hypertext systems lack the dynamic and general characteristics inherent in a DSS shell. The heart of this research is our model of *generalized hypertext*, which automates and generalizes the standard notion of hypertext for a knowledge-based DSS shell. Generalized hypertext superimposes a hypertext *network* on a DSS application, generating all nodes, links and link markers dynamically from the application's standard, non-hypertext knowledge base. Happily, this occurs unbeknownst to DSS application builders and users; they can take the hypertext-style interface for granted (see Figure 3). We have implemented many of these concepts in a prototype system called Max, which is used by the U. S. Coast Guard [KPBB90].

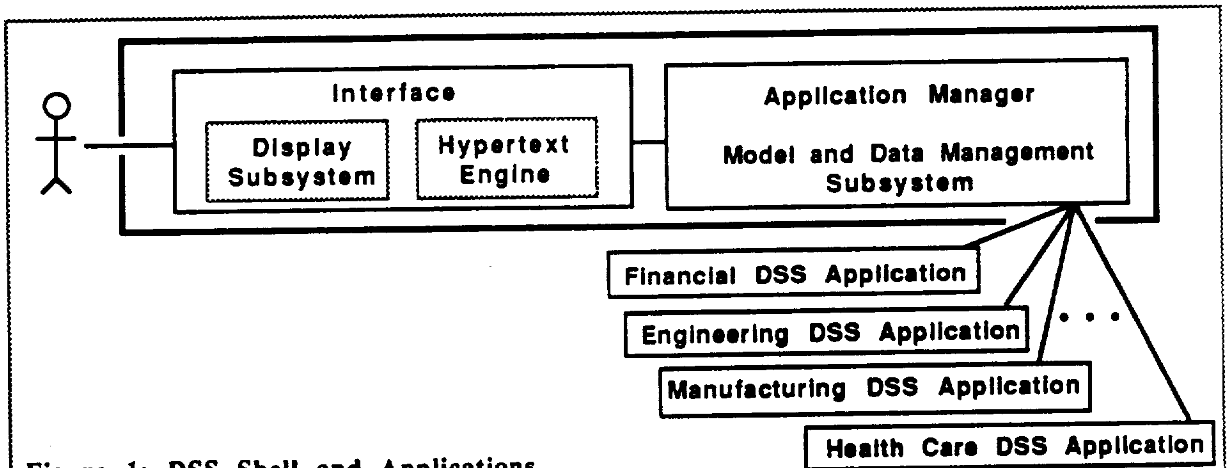


Figure 1: DSS Shell and Applications

The purpose of this paper is to discuss generalized hypertext and show how automating hypertext can enhance decision making with a DSS. We shall introduce generalized hypertext in §3 and show how it overcomes

[1] Shells provide standard functionality and a common "look" to a range of applications, with the goal of decreasing the effort involved in building and using them. For example, spreadsheet packages such as Lotus 1-2-3 can be considered shells. They provide a standard interface (the tabular worksheet), a set of functions (e.g., statistical, financial, as well as plain arithmetic) and set of menu commands. The applications are the individual spreadsheets built by entering formulas and data values in the interface provided. For a further discussion of shells see [Kim86, KPBB90].

Figure 2a: A hypothetical *interactive document* showing the results of executing the "asset" model from a DSS application. Interactive documents provide the following functionality. All text highlighted in boldface are hypertext *link markers* indicating the presence of hypertext *links* to documents and other related information. In this figure the user has selected two markers, the "asset" model and the execution result "$316.6m." The system then displays all appropriate commands and connections to information related to these markers, e.g., describe the selected model, execute the selected model, explain the selected model execution result. Each of these options represents a hypertext link. The user then may choose one of these links to *traverse*. In this illustration the user chooses an "explanation" link for the marker "$316.6m", i.e., option (1). We continue this scenario in Figure 2b.

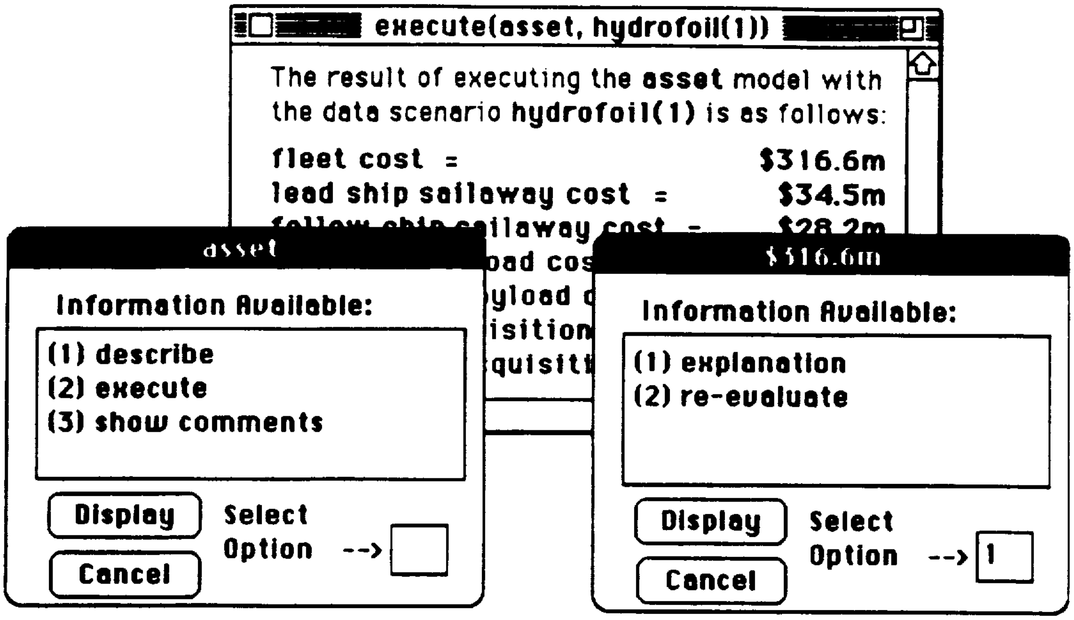
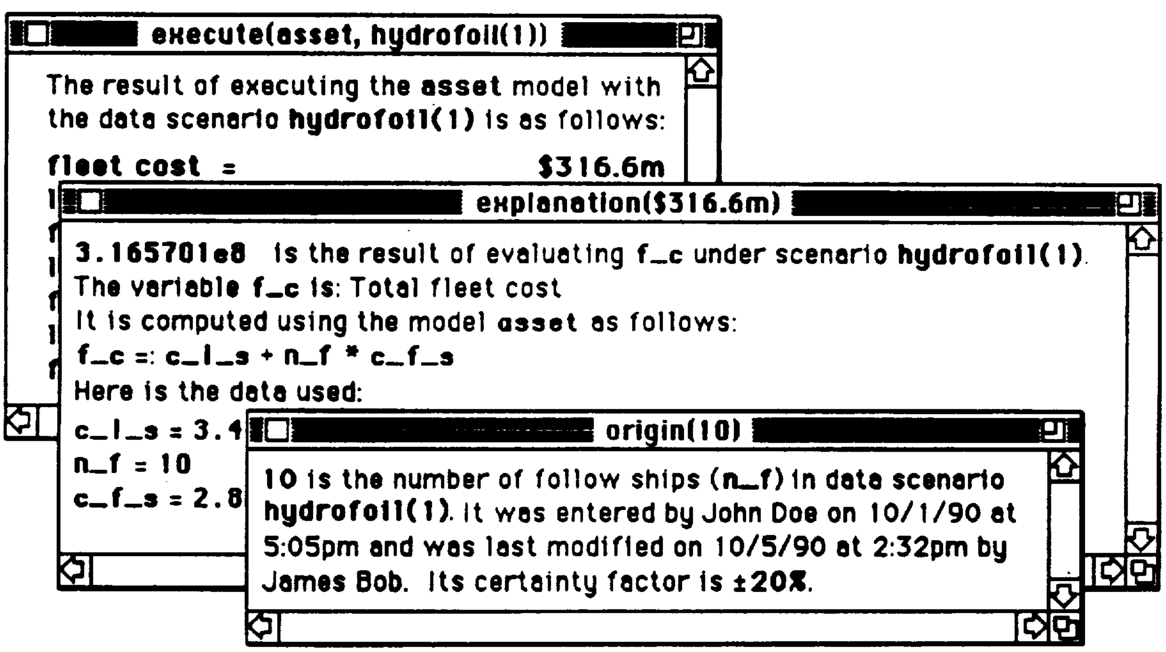


Figure 2b: A hypothetical example of *drilling down* for further detail in a DSS. In Figure 2a the user selected the link marker "316.6m" and chose to *traverse* an "explanation" link. The system displayed the resulting explanation shown here in the middle interactive document report. Then (s)he selected the link marker "10" in the second to last line. One of the available options was to show the origin of this value. The user chose this and the system *traversed* its corresponding link. This creates the interactive document report entitled "origin(10)".



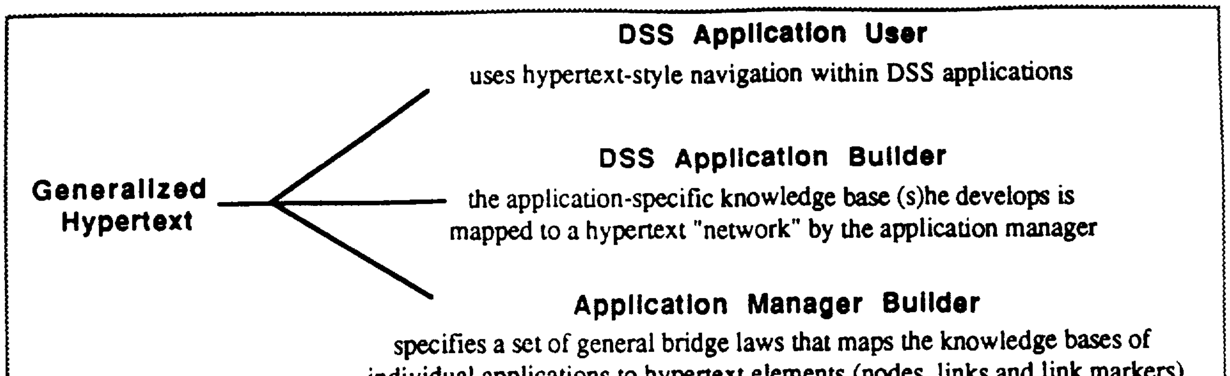


Figure 3

limitations inherent in what we call *standard hypertext*, the way hypertext traditionally has been implemented. First, in §2 we set the stage by explaining how a hypertext-oriented DSS can support decision makers.

§2 Hypertext and Decision Support Systems: Supporting the Four Stages of Decision Making

In this section we describe how DSS applications can assist decision makers and how hypertext can augment DSS functionality. To do so we shall use a framework comprising Herb Simon's *intelligence-design-choice* model of decision making [Sim77[2]] and Henry Mintzberg et al.'s additional step, *authorization* [MRT76 p257]. *Intelligence* is the act of gathering information. *Design* involves developing alternative scenarios or problem solutions. *Choice* is selecting the "best" option. *Authorization* is the process of justifying the recommendation or decision made to those affected by it.[3] In the paragraphs that follow we discuss the role for DSS and hypertext in each stage of decision making and justification.

Intelligence

Intelligence is the act of exploring a decision domain, either looking for or responding to a problem or opportunity. A DSS can help by maintaining information—the models[4], data and reports pertinent to understanding the general decision domain and, in particular, the issue-at-hand. For example, for a mortgage domain the DSS would contain information about the economy, interest rates and the housing market. Its models would help a loan officer evaluate a client's risk factor and determine what rate to charge. In an acquisition domain the DSS would contain life cycle cost models (e.g., construction, operation and maintenance equations) as well as exact or statistical data on the items being purchased or constructed. Similarly, scientific and engineering domains would have their own sets of models, data and standard reports.

Not only should the DSS maintain these models and data, but it should help the decision maker (or the analyst working for him) access and understand these. Now consider hypertext, which many believe reduces the

---

2 For an interesting analysis see [Sil86 p24].

3 Of course, instead of being performed sequentially, these stages typically are intertwined during a decision analysis.

4 There are many types of decision models: algebraic (e.g., sets of formulas found on a spreadsheet), optimization (e.g., sets of equations used in linear and integer programming), simulation, etc. Models are *executed* by applying a *data scenario*—a set of data values—to the model's variables.

cognitive overhead in viewing a complex domain [Con87 p40][5]. We can think of hypertext as a method for presenting short information displays embedded with links to further details or other related information. Hypertext-style *navigation* or *browsing* allows the user to explore this "network" selectively, choosing to see what he wants to and by-pass what he already understands or feels is less important at the moment. What would we want to link in a DSS? Related models, data and reports should be connected. Models should be linked to their submodels and variables. These variables should be linked to all possible data values registered in an application data base. All of the above should be linked to any execution results derived from them, especially when these results appear in reports. The user should be able to retrieve definitions, explanations and any background information (including any comments or annotations) inferable about items of interest. We illustrate access to such information in Figures 2a and 2b. The DSS interface can use hypertext concepts when structuring and presenting information, so that relevant data (e.g., based on these types of links) is readily accessible to the analyst or decision maker in a comprehensible and useful format [HM89 p47]. We have dubbed this hypertext-style accessibility the *WYWWYWI ("what you want, when you want it") principle* [BBK88].

Design

The analyst will design potential scenarios to address the issue-at-hand. This may involve creating additional models or developing alternate data scenarios (sets of data values for the model equation variables), perhaps through spreadsheet-style "what-if" analysis. For example, an analyst in the acquisition domain may explore the effects of possible changes in petroleum prices or the inflation rate on operation and maintenance costs over a thirty-year period.

To support the hypertext-style navigation described earlier, the DSS must both register the new models and data scenarios, and create the hypertext network—the nodes, links and link markers accessed during exploration and execution.

Choice

According to the *principle of decision support* [Kim88], the analyst will use the DSS to investigate different alternatives until he comes up with an optimum choice or quits (e.g., for timing reasons or perhaps because there is no optimum), in which case he will choose the best alternative he has found (assuming it is "satisfactory"). Normally the DSS's operations include executing models and performing *what if* analysis.

How does hypertext help? Again, hypertext provides easy access to explanations and other information. Furthermore, hypertext-style markers can act as *embedded menus* [KS86], giving the analyst "context-sensitive" access to normal DSS operations, such as model execution. We see this in Figure 2a.

Authorization

Analysts often use a DSS to develop a recommendation. The tangible result of a completed DSS analysis is a report documenting it. Decision makers can use this report to reach a final decision and then convince others of its logic.

For example, Figure 4 shows the final report of a procurement committee. Hypertext's role would be to facilitate the connection between this report and all supporting information linked to the highlighted markers. This should help the decision maker in several ways. The DSS (ideally) would contain the answer to any questions he would have, especially if the analyst has added comments. The decision maker can investigate any piece of information, exploring all the way "down" to its origin (recall Figure 2b), thereby increasing his overall understanding of, and confidence in, his decision. The most important point is that the decision maker is now self-sufficient. He can do all this by browsing on the computer without active assistance from the report's author.

---

[5] At its most basic level, *hypertext* is simply the concept of *linking* any two pieces of information, for example, a number with an explanation of how it was generated. For surveys of hypertext history and systems see [Con87, SK89 and Nie90]. [IRIS89 and Nie89] provide further bibliographic references. See [Min89] for a general discussion of hypertext and decision support systems.

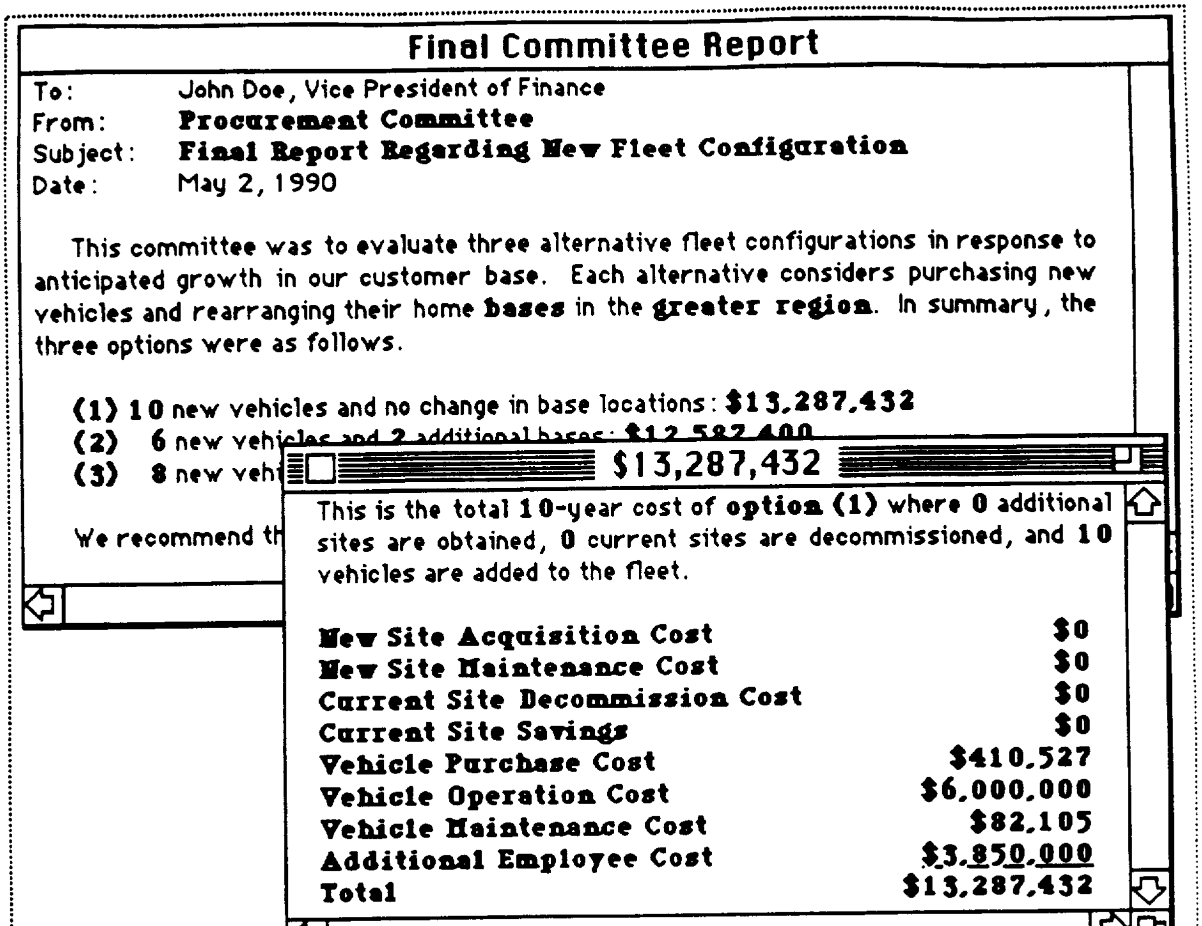


Figure 4

In summary, these stages implement the *argumentation theory of DSS*, according to which "the main purpose of a DSS is to support the construction, evaluation and comparison of arguments for courses of action" [Kim88, KPBB90]. These "arguments" then are used to justify the course of action taken. The hypertext concepts employed by the interface facilitate such support, giving the user easy access to information and operations within the DSS application without overwhelming him with details.

Can this process be made more efficient? Consider the second interactive document in Figure 4, which presumably had to be constructed manually. Suppose the system could generate this report, along with all other lower-level definitions and explanations. This would expedite the analyst's task dramatically! It is this DSS functionality that we shall describe in the next section.

§3 Generalized Hypertext: Overcoming the Limitations of Standard Hypertext

The DSS environment described in §2 could be supported by most hypertext systems today, but with a great restriction in functionality. To illustrate this, in Technical Appendix 1 we give the internal code for part of the knowledge base defining the interactive documents shown in Figure 2b. (A systems programmer would arrange such code based on the nodes, links and link markers created by the application builder, the application builder and user never see this view of hypertext[6].) What is wrong with this internal representation? The problem is that the knowledge base consists of static, explicit, pre-defined entries. The builder must anticipate the user and provide all the nodes, links and markers he believes the user will want to access in the future. He therefore must do all DSS analyses and prepare their explanations in advance so he can specify the hypertext link markers embedded in the appropriate reports. Each application builder must establish the hypertext nodes, links and markers related to his application.

Considering all the possible executions, execution results and explanations hinted at in Figures 2 and 4, we see that in a typical application, an immense amount of analysis is possible, too much to do in advance. Furthermore, it would be unreasonable to ask the application builder to declare all the hypertext elements in his application. The effort in doing a thorough job would frighten away the most willing application builder. Besides, a DSS generally is a dynamic beast. Its knowledge base contains definitions, models, data and documents, but not execution results or their explanations. The latter are produced dynamically upon user request, and so must any hypertext links mapped to them. Our only feasible option is to automate the generation of the hypertext network.

How do we automate hypertext in a knowledge-based DSS? First, it is important to note that hypertext functionality (e.g., creating, exploiting and managing links and linked information items) must be implemented in a way that is accessible to all current and future shell applications. Therefore we treat hypertext functionality as *system-level* (as opposed to application-level) support. Recall Figure 1, where the hypertext engine was embedded in the shell's interface component. Like database systems and user interface management systems, the hypertext engine is an application-independent, system-level tool for providing hypertext functionality for specific applications.

When automating hypertext, the hypertext engine cannot take an arbitrary application's knowledge base and magically infer what elements correspond to hypertext nodes and links. Instead, the shell's *application manager* subsystem must provide some translation routines that the engine can use to make its inferences[7]. We call these translation routines *bridge laws* [Nag61, Hau78, Kim79] because they serve as a "bridge" or connection between elements defined in the language of the application's knowledge base and those in the shell's hypertext engine. As we shall see, bridge laws exploit *logical quantification*, enabling individual laws to map entire classes of application objects (e.g., models or variables) to hypertext entities; the same bridge law will map any number of instances in the application knowledge base. For example, recall the three possible links emanating from the "asset" model in Figure 2a—"describe", "execute" and "show comments." How could this set be generated in a general manner? Given any link marker that represents the name of a model, in Technical Appendix 2 we describe a set of eight *predicates* that generates this trio of links. These predicates will work for every model ever to be declared by an application builder, now and in the future. In [Bic90] we describe similarly compact sets of bridge laws for variables, data scenarios, etc. Additional bridge laws may be developed by the systems programmer responsible for the shell's application manager subsystem. Application builders and users need have no knowledge of bridge laws. To them, hypertext functionality occurs automatically!

In developing generalized hypertext we had to solve three of the outstanding problems in hypertext research as identified by Halasz in [Hal88]. These are the creation and manipulation of virtual hypertext entities, computation over the knowledge base during link traversal, and the tailorability of the hypertext network. We describe these in

---

[6] The shell's interface is much "friendlier" than this code. For example, in a typical hypertext system, an application builder would construct a link by selecting a portion of text with his mouse, choosing the "create link" menu, and selecting a second portion of text in a different window. The system would then generate the internal code for the link, similar to that shown in Technical Appendix 1.

[7] Analogous to a spreadsheet package that provides all necessary functions and menu commands to an individual spreadsheet, a DSS shell *application manager* subsystem would provide both the standard report templates and the tools for manipulating models and data (e.g., execution and explanation), as long as application builders declare their models and data in a standard format, much as one is forced to in a spreadsheet. Based on this standard format, the application manager in a hypertext-oriented DSS shell provides general bridge laws for *mapping* application models, data, template reports, functions and commands to hypertext nodes, links and link markers.

the following paragraphs.[8]

Bridge laws are examples of *virtual entities*[9]. By *virtual*, we mean not fully identified or *resolved*. The parameters of a virtual entity are not known or *instantiated* in their entirety and therefore can assume any compatible value. For example, the "316.6m" link marker in Figure 2a is virtual when the user selects it because two possible links can be associated with it, i.e., its "link" parameter can be filled by one of the two compatible links. One can think of virtual entities being represented by templates prescribing both the entity's internal format and how the hypertext engine should *infer* compatible values to fill its parameter *slots* during link traversal. (We describe the actual technique used in Technical Appendix 2.) When the hypertext engine has filled all the slots then we say that the virtual entity is *resolved* and that we have either found or created (generated) a specific *instance* of that entity. When the user selects the "316.6m" link marker, the hypertext engine uses the bridge laws for link markers and links to *infer* which links could exist for the markers selected and tentatively fills in the templates with values drawn from models and data in the application knowledge base. The user then chooses one of these tentative links to traverse, as shown in Figure 2a. As Appendix 2 indicates, several pieces of information now must be inferred to resolve the link chosen. One is the type of link (e.g., "explanation" or "re-evaluation"). Another is the link's destination. The hypertext engine uses additional bridge laws to infer these from the contents of the application knowledge base. Once these are known, we consider the link fully resolved because the template is complete.

The next step is *computation*, i.e., actually generating the contents of the destination node specified by the link. A DSS operation (e.g., a model execution) normally must be performed to create the destination node (e.g., a decision report describing the results of the operation). Once this is done, the shell prepares the outcome for display in an interactive document. Part of this process involves the hypertext engine, which uses bridge laws to determine which portions of the interactive document should be highlighted as link markers. These markers will be virtual, for as we saw, they will not be associated yet with actual links. When the user selects one of these markers, the resolution cycle repeats. In summary, we say that link traversal in generalized hypertext follows the *select-infer-traverse-infer* pattern described here.

We want an application to be able to *tailor* its processing and resultant reports to a given user's skill level and the specific task he is using the DSS application to accomplish. The shell can do this by maintaining multiple *contexts* or modes, each associated with a different *view* of application knowledge base components and with a different set of report formats. The hypertext engine checks contexts as part of every operation it performs. For specific details we refer the reader to [Bie90]. Contexts are also instrumental in providing the task environments discussed in the next section.

§4 Discussion and Future Developments

Generalized hypertext is a logic-based technique for automating hypertext within a knowledge-based decision support environment. Generalized hypertext adds value by providing a hypertext-style interface to a decision support system (DSS) application without an *author* having to create any nodes or links. (Of course, users can add their own comments and other annotations, just as in regular hypertext systems.) Generalized hypertext can be applied to any domain that is well enough understood and expressible for the systems programmer building an application manager subsystem to map the components of the application knowledge base to generalized hypertext entities and attributes.[10]

The direct access to information provided by a generalized hypertext DSS makes its use easy for someone who is new to a decision domain. He can explore the domain at his own pace, read the comments and definitions, and

---

[8] We discuss this entire process more fully in [BK90] and [Bie90].

[9] By hypertext entities we mean hypertext nodes, links and link markers. Further hypertext entities are presented in [Bie90].

[10] For an interesting discussion of domains where this is not possible see [RT88].

experiment with application models and data. Another benefit of hypertext is the degree of user control. More advanced users can by-pass information easily with which they are familiar. To assist the user further, one direction we shall explore in our research is *task environments*, local environments organized around the procedure the analyst should follow when he is performing an individual task [Bie90]. They are tailored specifically to the task (and perhaps to the user's skills or preferences). Only relevant information (data, documents, reports, models) and commands will be available. Also, task environments can serve as an enhanced form of documentation, building upon the notion of, e.g., *guided tours* in NoteCards [MI89] or Zellweger's *scripted documents* [Zel89]. An advanced implementation of task environments could incorporate "active" agents that guide the user through the process, reminding him of steps left out and making suggestions.

Generalized hypertext enhances the functionality and ease of use of DSS applications within a DSS shell. This is evident from our prototype system, Max, described in [KPBB90]. It is our belief that generalized hypertext will advance both the state of hypertext research in knowledge-based environments and the art of information presentation in decision support systems.

Acknowledgement

I want to thank Steven O. Kimbrough of the Wharton School, University of Pennsylvania for his help in developing generalized hypertext, which was invaluable.

This work was motivated by and funded in part under contract DTCG39-86-C-80348, between the U. S. Coast Guard and the University of Pennsylvania, with Steven O. Kimbrough as principal investigator.

Technical Appendix 1: Inside a Standard Hypertext Knowledge Base

This appendix presents a systems programmer's view of the nodes, link markers and links in the hypertext knowledge base representing Figure 2b's reports. Each report, link marker and connection is represented by a "node", "marker" or "link" *predicate*. We precede each set of predicates by a legend explaining its three arguments.

Legend: node(unique report id, report title, list of the text and link markers comprising its contents)

```
node(1, title("execute(asset, hydrofoil(1))"),
    content(["The result of executing the", marker(1), "model with the data scenario", marker(2), "is as
        follows:<cr>", marker(3), "=", marker(4), "<cr>", marker(5), "=", marker(6), ...]))
node(2, title("explanation($316.6m)"),
    content([marker(17), "is the result of evaluating", ...]))
node(3, title("origin(10)"), content(["This value for the number of helicopters..."]))        ... etc.
```

Legend: marker(unique marker id, the link associated with the marker, the text representing the marker)

```
marker(1, link(1), content("asset"))
marker(2, link(2), content("hydrofoil(1)"))
marker(3, link(3), content("fleet cost"))
marker(4, link(4), content("$316.6m"))        ... etc.
```

Legend: link(unique link id, the report where the link originates, the link's destination report)

```
link(1, node(1), node(4))
link(2, node(1), node(5))
link(3, node(1), node(2))
link(4, node(1), node(3))        ... etc.
```

Technical Appendix 2: Generalized Hypertext

This appendix presents simplified examples of bridge laws, which are specified using quantified predicates in first order logic. The set of predicates here, which includes both bridge laws and some of the application knowledge base declarations they find, generates the hypertext links emanating from all keywords in interactive documents that are names of application models. For clarity, we precede bridge law names with the code "ght" and application knowledge base declarations with "appl".

Legend: The format of the generalized hypertext link predicate

We declare generalized hypertext links with the following predicate. As we shall see next, bridge laws for links have this same format.

```
ght_link(id, originating node, originating marker, destination node, link operation type)
```

Bridge Law Declaring Links from Keyword Markers to Destination Nodes

This bridge law establishes links to all keywords *i* that are names of elements in the DSS application. It calls the application predicate *appl_type/2* to identify the elements, bridge law *ght_operation/2* to determine the link operation type *o*, and bridge law *ght_identifier/3* to generate the name of the destination node *x* given the link operation identified. (Arguments with underscores accept any values passed.)

```
(∀i, o, t, x) (ght_link(i, _, keyword(i), x, o) <--
    (appl_type(i, type(t))  &
     ght_operation(type(t), i, o)  &
     ght_identifier(i, o, x)))
```

Identifying the Type of Application Element Selected

It is reasonable to assume that applications can identify each of their elements. This predicate determines that *i* is an application model if the application has an *appl_model/3* predicate declared for it.

```
(∀i) (appl_type(i, type(model)) <--
    (appl_model(i, _, _)))
```

Bridge Laws Inferring Link Operation Types for Models and Elements with Comments

The first two bridge laws find "describe" and "execute" links for application models.

```
ght_operation(type(model), _, operation_type(describe))

ght_operation(type(model), _, operation_type(execute))
```

The next determines whether any comments are registered for the element passed.

```
(∀i) (ght_operation(_, i, operation_type("show comment"))  <--
    (comment(i, _)))
```

Bridge Laws Inferring Destination Node Identifiers

Given the link operation type, these laws infer the identifier of the destination node that the link operation will generate. The destination identifier is passed in the third argument.

```
(∀i) (ght_identifier(i, operation_type(describe), describe(i))  <--
    (appl_type(i, type(model))))

(∀i, d) (ght_identifier(i, operation_type(execute), execute(i, d))  <--
    (appl_type(i, type(model))  &
     appl_data_scenario(i, d)))

(∀i) (ght_identifier(i, operation_type("show comment"), comment(i)))
```

References

[BBK88] Bhargava, Hemant K., Michael P. Bieber and Steven O. Kimbrough, "Oona, Max, and the WYWWYWI Principle: Generalized Hypertext and Model Management in a Symbolic Programming Environment," in *Proceedings of the Ninth International Conference on Information Systems*, Minneapolis, 1988, pages 179-192.

[Bie90] Bieber, Michael, Generalized Hypertext in a Knowledge-based DSS Shell Environment, Ph. D. Dissertation, Decision Sciences Department, University of Pennsylvania, 1990.

[BK90] Bieber, Michael and Steven O. Kimbrough, "Towards a Logic Model for Generalized Hypertext," *Proceedings of the 23nd Hawaii International Conference on System Sciences*, Hawaii, 1990.

[Con87] Conklin, Jeff, "Hypertext: a Survey and Introduction," *IEEE Computer*, volume 20 number 9, 1987, pages 17-41.

[Hal88] Halasz, Frank G., "Reflections on Notecards: Seven Issues for the Next Generation of Hypermedia Systems", *Communications of the ACM*, volume 31 number 7, July 1988, pages 836-855.

[HM89] Herrstrom, David S. and David G. Massey, "Hypertext in Context," in The Society of Text: Hypertext, Hypermedia, and the Social Construction of Information, Edward Barrett (ed), MIT Press, Cambridge, MA, 1989, pages 45-58.

[IRIS89] "Hypermedia Bibliography," Institute for Research in Information and Scholarship (IRIS), Box 1946, Brown University, Providence, RI 1989.

[Kim79] Kimbrough, Steven O., "On the Reduction of Genetics to Molecular Biology," *Philosophy of Science*, volume 46 number 3, September 1979, pages 389-406

[Kim86] Kimbrough, Steven O., "On Shells for Decision Support Systems," Wharton School, Department of Decision Sciences, working paper #86-07-04, July 1986.

[Kim88] Kimbrough, Steven O., "The Argumentation Theory for Decision Support Systems," draft of Chapter 2 of Decision Support Systems: Technologies for Reasoning and Argumentation, Wharton School, Department of Decision Sciences, 1988.

[KPBB90] Kimbrough, Steven O., Clark W. Pritchett, Michael P. Bieber and Hemant K. Bhargava, "An Overview of the Coast Guard's KSS Project: DSS Concepts and Technology," *Transactions of DSS-90*, TIMS, Boston, MA, May 21-23, 1990, pages 63-77; accepted for publication in *Interfaces*.

[MI89] Marshall, Catherine C. and Peggy M. Irish, "Guided Tours and On-Line Presentations: How Authors Make Existing Hypertext Intelligible for Readers," in *Proceedings of the 1989 Hypertext Conference*, Pittsburgh, PA, November 1989, pages 15-42.

[Min89] Minch, Robert, "Application and Research Areas for Hypertext in Decision Support Systems," *Journal of Management Information Systems*, volume 6 number 3, Winter 1989-90, pages 119-138.

[MRT76] Mintzberg, Henry, Duru Raisinghani and André Theoret, "The Structure of Unstructured Decision Processes," *Administrative Science Quarterly*, volume 21, June 1976, pages 246-275.

[Nag61] Nagel, Ernest, The Structure of Science: Problems in the Logic of Scientific Explanation, Harcourt, Brace &World, Inc., New York, NY, 1961

[Nie89] Nielsen, Jakob, "Hypertext Bibliography," *Hypermedia*, volume 1 number 1, Spring 1989, pages 74-91.

[Nie90] Nielsen, Jakob, Hypertext and Hypermedia, Academic Press, 1990.

[RT88] Raymond, Darrell R. and Frank W. Tompa, "Hypertext and the Oxford English Dictionary", *Communications of the ACM*, volume 31 number 7, July 1988, pages 871-879.

[Sim77] Simon, Herbert, The New Science of Management Decision, Harper and Row, NY, NY, 1960, 1977 (revised edition).

[Sil86] Silver, Mark S., Differential Analysis for Computer-Based Decision Support, Ph. D. Thesis, Wharton School, 1986.

[SK89] Shneiderman, Ben, and Greg Kearsley, Hypertext Hands-On! An Introduction to a New Way of Organizing and Accessing Information, Addison-Wesley, Reading, MA, 1989.

[Zel89] Zellweger, Polle, "Scripted Documents: A Hypermedia Path Mechanism," in *Proceedings of the 1989 Hypertext Conference*, Pittsburgh, PA, November 1989, pages 1-14.

Session 7

Future of Hypermedia

Chair: David Palumbo

Hypermedia as Medium

Chris Dede

Hypermedia = Hypercommunication

Mark R. Laff

Moving from Knowledge Presentation to Knowledge Representation

David Palumbo

N91-21952

Hypermedia As Medium

This Working Paper is an early partial draft of
an article for the Hypertext Special Issue
of the *International Association for Impact Assessment Bulletin*

Dr. Christopher Dede, Director
Advanced Knowledge Transfer Project
University of Houston—Clear Lake
Houston, Texas 77058

HYPERMEDIA AS MEDIUM

Abstract

This paper describes claims and rebuttals that hypermedia (the associative, nonlinear interconnection of multimedia materials) is a fundamentally innovative means of thinking and communicating: a new medium. Advocates of hypermedia pose several arguments for why this representational architecture is a major advance over other media:

- the associative, nonlinear nature of hypermedia mirrors the structure of human long-term memory
- the capability of hypermedia to reveal and conceal the complexity of its content lessens the cognitive load on users of this medium
- the structure of hypermedia facilitates capturing and communicating knowledge, as opposed to mere data
- hypermedia's architecture enables distributed, coordinated interaction, a vital component of teamwork

However, skeptics argue hypermedia has several intrinsic problems that severely limit its effectiveness as a medium:

- people become disoriented when navigating through large hypermedia structures
- traversing a hypermedia network imposes considerable cognitive overhead on the user
- creating hypermedia structures involves a very large front-end investment of time and expertise
- "Tower of Babel" situations are likely in shared hypermedia systems

Few technological innovations have greater potential to transform society than does a new medium (e.g. the long-term effects of the invention of writing). If hypermedia does transcend its limits to infuse civilization as a new means of thinking and communicating, the consequences will be profound.

Introduction

Since the dawn of civilization, few full-fledged media have emerged as vehicles for thought and interaction. The spoken word, the written word, still images, and full-motion images are the major representational methods that people have evolved to symbolize and communicate ideas. Enthusiasts for hypermedia (the associative, nonlinear interconnection of multimedia materials) are now claiming that a new medium has emerged. If this assertion is true, then a host of direct and indirect consequences for civilization will follow. Few technological innovations have greater potential to transform society than a new medium (e.g. the long-term effects of the invention of writing). This paper describes claims and rebuttals that hypermedia is a fundamentally innovative means of thinking and communicating.

All media except the spoken word have required a technological infrastructure; their advent in history coincided with the development of innovations capable of actualizing such a medium. Full-motion images, for example, were not widespread until a cluster of technological advances in the early part of this century empowered "motion pictures" (both photographic and animated). Hypermedia has not emerged until the present because powerful computers with relatively large memories are needed to create, store, and display large webs of nodes and links. The usage of computers has altered almost every aspect of society; will hypermedia be the next major impact, the first computer-centered medium?

Reasons for Excitement about Hypermedia

Advocates of hypermedia pose several arguments for why this representational architecture is a major advance over other media:

- the associative, nonlinear nature of hypermedia mirrors the structure of human long-term memory, empowering both intelligence and coordination through intercommunication
- the capability of hypermedia to reveal and conceal the complexity of its content lessens the cognitive load on users of this medium, thereby enhancing their ability to assimilate and manipulate ideas
- the structure of hypermedia facilitates capturing and communicating knowledge, as opposed to mere data
- hypermedia's architecture enables distributed, coordinated interaction, a vital component of teamwork, organizational memory, and other "group mind" phenomena

The intuitive case for each of these arguments is presented below; the section following describes reasons for skepticism about these claims.

Hypermedia is Associative and Nonlinear: Human long-term memory is not similar to a computer database; remembering something does not involve doing a pattern-matching search through large numbers of records. Instead, the information a person assimilates is organized into an elaborate web of associations. For example, the term "apple" conjures up connotations of pies, orchards, Isaac Newton, the Beatles, the Garden of Eden, and a computer corporation—all interrelated through an correlational network. Our adeptness in quickly storing and retrieving large amounts of information seems to stem from this property of associativity (Caudill, Butler, 1990).

One of the most important subdisciplines in artificial intelligence is knowledge representation. This field centers on devising formalisms that allow a computer's logical mechanism to access and manipulate knowledge (a more complex entity than data, as will be discussed later). Some of the most useful representational architectures researchers in this area have developed—frame-based expert systems, object-oriented databases, semantic networks, and hypermedia—all share the property of associativity.

From a different intellectual perspective, biological researchers are studying neural networks (the physiological infrastructure underlying associative memory) to determine why organisms have such powerful abilities to recognize complex patterns and to learn. This work is generating insights about how simultaneous, distributed processes can be coordinated through intercommunication. From these studies of associative physiological networks, many applications are emerging in fields ranging from manufacturing to cognitive science. Research from these multiple perspectives suggests that associativity is fundamental to both to intelligence and to coordination via communication.

Hypermedia is much more associative than traditional media because of its nonlinearity. Spoken and written speech, still and full-motion images are all linear media: each conveys a sequential stream of data. Packets of information based on these media have a beginning, a middle, and an end; authors seek to find a single logical flow that expresses the totality of ideas they wish to communicate to their audience. Each concept is locationally associated only to those ideas preceding and following it in the linear stream.

In contrast, nodes in hypermedia can have an arbitrarily large number of direct associations (links) to other concepts. This flexibility is valuable in generating and communicating ideas. For example, divergent thinking (e.g. brainstorming) is difficult to capture when the ideas must be summarized convergently in quasi-linear form (i.e. a hierarchical outline). Mental models are have richer associations than can be represented by the "tree" of correlations a hierarchical structure can express; hypermedia provides a formalism that can depict this complexity.

As another illustration, recontextualization (seeing a phenomenon from a variety of perspectives) is a powerful approach to problem solving. Using this approach requires a medium capable of simultaneously representing multiple mental models (in the same way that a two-dimensional optical illusion can be interpreted as different solid objects, depending on the virtual orientation selected by the viewer). Hypermedia's flexibility as a representational formalism facilitates recontextualization; a web, for example, can be conceptualized via analysis (as a set of nodes) or synthesis (as a network of links). The combination of associativity and nonlinearity in hypermedia adds dimensions to thought and communication lacking in other media.

Using Hypermedia Lessens Users' Cognitive Load: Because hypermedia mirrors the representational architecture of long-term memory, assimilating and communicating thoughts in this format may require less internal preprocessing. The effort involved in translating to and from an associative intellectual structure is eliminated. For example, authors can develop writer's block even when they know everything they wish to express. Part of the problem may the challenge of mapping long-term memory's associative web of relationships into a linear stream. Hypermedia could serve as an external, virtual mirror for a person's memory: A writer could follow different trails through a web instantiating his knowledge until he found the best linear path to express the ideas in his document.

Beyond this, hypermedia seems a powerful approach for revealing and concealing complexity when analyzing a phenomenon. As an illustration, abstraction is an important cognitive strategy, but thinking abstractly involves rapidly shifting among different levels of specificity. Problem solving often requires moving from an overall perspective through increasing amounts of detail until an insight occurs and a subproblem less complex than the total situation is mastered. Then this understanding is generalized, as much as possible, to test its utility in comprehending other aspects of the problem.

Hypermedia can empower this type of intellectual process by providing ways to represent and navigate through complex levels of abstraction. Detail can be concealed until needed, then revealed by activating a link. A node, when viewed at a greater level of specificity, can reveal a subnetwork of nodes and links as its internal structure: webs embedded within webs. Attempts to create similar representations in the medium of paper generate a useless spaghetti of lines.

People using hypermedia report that this medium seamlessly interconnects its contents. Creating regularities in the menu structure of different tools on a computer is an important design strategy; users want the same commands to activate the "delete" function whether they are working with a word processor, a database, or a spreadsheet. In the same manner, consistency across the interface between the thinker and the subject of thought reduces cognitive load. The multiple representational formalisms hypermedia supports are all accessed in the same manner (activate a link). This ease of use stands in sharp contrast to a person juggling an atlas, a database of still images, an encyclopedia, a cassette tape, and a videoplayer to acquire information.

Hypermedia Facilitates Capturing and Communicating Knowledge: Because of its associativity and low cognitive load, hypermedia is an attractive representational architecture for knowledge bases. "Data" can be defined as input gathered through the senses, and "information" as a pattern of input that signals an important change in the environment. In this schema, "knowledge" is integrated information that can be used to achieve a goal. To illustrate these definitions, learning that a new type of workplace tool exists would be data, realizing that it could add valuable functions to the one's occupational repertoire would be information, and mastering the tool would be knowledge.

Past generations of information systems have used advances in hardware and software to augment users' access to data, on the assumption that individual and institutional knowledge would thereby increase. After several decades of advances in data processing, however, even personal computers can deliver so much information that their users become overwhelmed: unable to decide which data is important or to interconnect new information with existing knowledge. Researchers in artificial intelligence believe that future information systems will use emerging increases in power not to create faster and larger databases, but instead to deliver "knowledge bases": contextually targeted, associationally interconnected data with embedded computational inferencing mechanisms (Brodie, Mylopoulos, 1986).

An interesting metaphor for knowledge processing is the concept of "cyberspace," a term that originated in a science fiction novel (Gibson, 1984). Cyberspace would be a real-time, online, multi-person virtual world in which, through ideas from scientific visualization, cognitive entities would take on tangible, sensory form to facilitate access and manipulation. This idea has captured the imagination of researchers in human factors, scientific visualization, gaming, computer-aided design, architecture, artificial intelligence, virtual realities, networking, computer-supported cooperative work, and hypermedia.

The design of cyberspace environments for depicting knowledge poses many research issues (Benedikt, in press). These include the costs and benefits of reifying information, space-time axiomatics in artificial realities, magic versus logic as principles underlying user actions, the presentation of the self in a virtual context for group work, the meaning of travel and action when translated from a physical to a symbolic domain, coordinate systems for (un)real estate, the form and meaning (semiotics) of data objects, three-dimensional user interface design, visual languages, alternatives to a spatiotemporal metaphor for virtual reality, and the architecture of multi-dimensional data spaces. (An entertainment-oriented cyberspace would create a different set of challenges—its users could interact, go on shared adventures, get married or divorced, start businesses, found religions, wage war, hold elections, construct legal systems, tailor their virtual physical appearances, and assume alternative personal identities and interpersonal styles.)

Developers of cyberspaces are using hypermedia as their representational structure; no other medium can support the complex knowledge architectures required. Similarly, knowledge base research is focussing on associative formalisms, such as hypermedia, because interrelation seems central in transforming information to knowledge. An illustration of the importance of associativity is the encyclopedia: civilization's attempt to encapsulate the total span of knowledge.

The chunks of data that are interrelated in an encyclopedia (a linear medium) are only those in the span of a single article; this segmented approach captures a small fraction of the total knowledge of the articles' authors. For example, to comprehend the role in history of the year 1842, one would have to scan every paragraph in the encyclopedia for events related to that period, then try to make sense of this jumble of data. A hypermedia-based encyclopedia, in contrast, could support webs of knowledge stretching across the full spectrum of articles: One could contrast an economist's perspective on the causes of the American Civil War with that of a slave or of an abolitionist. An associative encyclopedia can represent multiple mental models for interpreting the same data: alternative knowledge structures.

Hypermedia Enables Distributed, Coordinated Interaction:
Hypermedia may make knowledge easier to communicate as well as easier to represent. In the emerging field of computer-supported cooperative work (less formally known as "groupware"), many researchers are using hypermedia as the representational formalism for their projects. Applications in computer-supported cooperative work (CSCW) center on seven themes:

- building shared mental models
- aiding group design and decision making
- developing machine-based organizational memories
- coordinating complex, multi-person tasks
- enabling collaboration despite barriers of distance and time
- reducing Information overload in organizations
- enhancing psychosocial interaction in machine-mediated communication.

Hypermedia's capabilities to capture and communicate knowledge—coupled with its low cognitive load—make it an excellent representational medium for these CSCW applications.

For example, the correlational nature of hypermedia is valuable in building organizational memories. Complex, long-term projects necessitate institutional knowledge bases that transcend the involvement spans of individual personnel; otherwise, important information is lost when people leave, retire, shift to another project, or simply forget. Hypermedia-based associative memories can recall information even when queries are incomplete or garbled, can store data in a distributed fashion, can detect similarities between new inputs and and previously stored patterns, and do not degrade appreciably in performance if some of the memory's components are damaged. These characteristics are very useful for a distributed, shared organizational knowledge base.

Educators also are finding that hypermedia can aid in coordinating learning and communicating knowledge. Dede (1990) describes how hypermedia-based applications from CSCW are changing the field of distance learning: The distributed, simultaneous processes underlying distance education can be orchestrated more effectively, and hypermedia's structure makes knowledge easier to transfer across barriers of distance and time.

Wenger (1987) has evolved a theory of knowledge communication, which he applies to artificial intelligence work on intelligent tutoring systems. Making knowledge communicable (the cognitive essence of instruction) requires many of the attributes hypermedia offers: easy translation into long-term memory, consistency across the interface between learner and subject matter, revealing and concealing complexity, support for multiple mental models, associativity. From the perspectives of both work and education, hypermedia seems a promising communications medium for distributed, coordinated interaction.

Potential Limits for Hypermedia

With all these potential advantages, enthusiasts argue, the world should become "hyperimmediated" as quickly as possible. However, skeptics argue hypermedia has several intrinsic problems that severely limit its effectiveness as a medium (Dede, et al., 1988). The major concerns currently being voiced about hypermedia are:

- people become disoriented when navigating through large hypermedia structures
- traversing a hypermedia network imposes considerable cognitive overhead on the user
- creating hypermedia structures involves a very large front-end investment of time and expertise
- "Tower of Babel" situations are likely in shared hypermedia systems

The intuitive case for each of these arguments is presented below; the next portion of this article summarizes the extent to which research supports these concerns.

Information that is organized in a complex manner poses a potential problem of user disorientation. In a linear medium, one can readily evaluate the extent to which a document's information has been traversed (how many pages read, how many left) and where a particular piece of data is located (chapter, section, paragraph). Large hyperdocuments may be more confusing. In a web of thousands--or millions--of nodes, how does one define a location in the network, establish a desired direction to move, or blaze a trail indicating those nodes already scanned? In a non-hierarchical structure, what type of coordinate system should be used to indicate where a piece of data has been stored? These problems of navigation and referencing are very challenging for networks with large numbers of nodes and links.

Even if a user familiar with a particular network experiences no disorientation, working in a hypermedia knowledge base entails some extra cognitive overhead. When entering material, an author must think carefully about how to link the information being added to the web which already exists. At each node they encounter, users must choose which link to follow from multiple alternatives and must keep track of their orientation in a complex multidimensional structure. The richness of a nonlinear representation carries a risk of potential intellectual indigestion, loss of goal directedness, and cognitive entropy. Unless a hypermedia system is designed carefully from a human factors perspective, increasing the size of the knowledge base may carry a cost of decreasing its usability.

The availability of multiple types of representations in a hypermedia system compounds this problem. Access to representational alternatives allows users to tailor input to their individual cognitive styles and enables authors to choose a format well suited to the material being entered. However, coping with multiple formats adds to cognitive overload, and little is known about which representational ecologies are functional for different task situations.

The large front-end investment in time and expertise required to author a large hypermedia structure is another type of potential limit. The challenges involved can be illustrated by considering the simple situation of adding a new node to an existing web. The number of links generated by adding an additional node to a network will vary depending on the type of knowledge being stored, the objectives of the documentation, and the sophistication of the user population.

Suppose that many vital, subtle interrelationships exist in the network's material. Although some new nodes will simply annotate single existing nodes, a substantial proportion of nodes that are added may require multiple links. In a million node web with 0.1% of the material interrelated, adding a single new node would require constructing one thousand. The difficulties of comprehending and maintaining such a web could exceed the benefits that a nonlinear medium provides.

One strategy for solving this problem is to aggregate subnetworks into composite nodes that chunk material on a higher level of abstraction: webs within webs. Such an approach is being explored in second generation hypermedia systems—but creating another dimension of hierarchy complicates the representational architecture and, unless implemented in a manner transparent to users, may increase disorientation and cognitive overload.

These potential limits are particularly acute in online, shared hypermedia systems. The user of a collegial electronic knowledge base may find that, since last entering the system, familiar paths have changed and new material has appeared. Links that seem intuitively obvious to the author adding them may be puzzling to others. Skeptics argue that "Tower of Babel" situations are likely with a large knowledge base in which multiple users can alter the fundamental medium of interaction.

Possible problems of disorientation, cognitive overhead, front-end investment, and collective communications dysfunctions reflect the intricacy of working with a knowledge base rather than a database. Knowledge is intrinsically complex, and transforming information to knowledge involves gaining a goal-directed, contextual understanding of the application domain. Utilizing an underlying representation based on hypermedia will require more sophisticated skills--a new type of "literacy"--from its users. Many are skeptical that the benefits of hypermedia will justify the costs of creating this hyper-literacy.

Conclusion: The (Hyper)Medium and the Message

Any medium shapes its message (and the recipient of that message). For example, the imagination and involvement of a child are engaged in different ways when reading a fairy tale in a book than when watching a movie depicting that story. The general question of how the medium shapes the message is emerging as a central issue for information technology research (Dede, in press).

For example, the primary goal of the discipline of telecommunications has been to improve the transmission capabilities of the various media while lowering their cost. Research has centered on technical issues underlying the electrical or optical transfer of information at a distance, irrespective of the content being conveyed. While the challenge of transferring signals reliably and cheaply is still an important theme, now the emphasis in studies of communications media is refocussing on how to extend a state of awareness and related intentions over a distance to others. The importance of this research for understanding the evolution of human thought and communication can be highlighted by describing two illustrative issues emerging in workplace and community settings.

In business, information technology is now seen as an effective method for improving worker productivity and for enhancing an organization's ability to respond quickly and flexibly to environmental changes. These gains are intrinsically accompanied by a decrease in face-to-face interaction and an increase in technology-mediated communication. Workers progressively have more and more colleagues with whom they exchange information, but their primary form of contact is through the telephone or electronic mail.

As a result, the psychosocial aspects of work are changing in complex ways that depend in part on the implementation strategy used to create the institution's information infrastructure. In general, people feel more productive, but less personally fulfilled by their work environment. The consequences of this shift for both workers and organizations are being studied, but our current understandings are very limited, and how the evolution of hypermedia might affect this situation is unclear.

The potentially troubling effects of using advances in technology, such as hypermedia, to mediate human experience are nowhere more clear than in today's community. The single greatest experiential input for many people is the pervasive sensory, informational, and normative environment created by television, radio, videogames, movies and videotapes. In this situation, people's knowledge and values can be constrained by the characteristics of these communications channels.

For example, concerns about "reality pollution" in the news media are mounting. Businesses produce and distribute self-serving, sometimes biased "docutainments" that the media broadcast to cheaply augment their programming. Images can now be doctored electronically to the point that their authenticity can no longer be determined. Political events are routinely followed by commentaries in which "spin control" experts attempt to skew viewers' perspectives on what they saw.

The cultural consequences of technology-mediated physical/social environments are mixed. On the one hand, people have a wider range of vicarious experience and more contact with specialized human resources than they could attain through direct interaction in their local region. On the other hand, to the extent that perceptions of family life come from situation comedies, of crime from police shows, and of sexuality from soap operas—and to the degree that physical exercise is confined to pressing the buttons on videogames—civilization is in serious trouble.

The technologies themselves do not dictate content that creates a "couch potato" mentality or implies all of life's problems can be solved in thirty minutes. As pervasive interpreters of reality, the media are influenced primarily by economic, political, and cultural forces. But, given that the medium does intrinsically shape the message, society must consider the extent to which communications technologies have generated passive, narcotic behavior for many of its members.

The factors that have created this situation and the potential impact of hypermedia on its dynamics are largely not understood. If hypermedia does transcend its limits to infuse civilization as a new means of thinking and communicating, the consequences will be profound.

References

Benedikt, M., Editor. (in press). Cyberspace: First Steps. Cambridge, Massachusetts: MIT Press.

Brodie, M.L. and J. Mylopoulos, Eds. 1986. On Knowledge Base Management Systems. New York: Springer-Verlag.

Caudill, M. and C. Butler. 1990. Naturally Intelligent Systems. Cambridge, Massachusetts: MIT Press.

Dede, C. (in press). Emerging Information Technologies: Implications for Distance Learning. Annals of the American Academy for Political and Social Sciences.

Dede, C. 1990. The Evolution of Distance Learning. Journal of Research on Computing in Education, 22, 3, 247-264.

Dede, C., T. Sullivan, and J. Scace. 1988. Factors Shaping the Evolution of Electronic Documentation Systems (RICIS IM-4). Houston, TX: Research Institute for Computing and Information Systems, University of Houston—Clear Lake.

Gibson, W. 1984. Neuromancer. New York: Ace Books.

Wenger, E. 1987 Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge. Los Altos, California: Morgan Kaufmann.

N91-21953

Hypermedia = Hypercommunication

Hypermedia '90
December 5, 1990
Houston, Texas

Mark R. Laff

Interactive Media Project
IBM T.J. Watson Research Center, Hawthorne
P.O. Box 704
Yorktown Heights, New York 10598
BitNet: MRL at YKTVMH
CSNet: mrl @ ibm.com

Abstract

New hardware and software technology have given application designers the freedom to use new realism in human computer interaction. High-quality images, motion video, stereo sound and music, speech, touch, gesture provide richer data channels between the person and the machine. Ultimately, this will lead to richer communication between people with the computer as an intermediary. The whole point of hyper-books, hyper-newspapers, virtual worlds, is to transfer the concepts and relationships, the "data structure", from the mind of the creator to that of the user. In this talk we will discuss some of the characteristics of this rich information channel followed by some examples of our work, including an interactive hypermedia biography of IBM Fellow John Cocke entitled "John Cocke: A Retrospective by Friends".

Introduction

For the author, one of the most "visual" pieces of classical music is Tchaikovsky's "Peter and the Wolf". The musical themes representing each of the characters add vivid imagery to the story. Readers of this article will each have their own favorite example of music which evokes powerful associations. Some of these links are cultural and shared, while others are private and unique to an individual's own experience.

Moving to another medium, the adage "a picture is worth 1000 words" can be modernized to "a moving video picture is worth 30,000 words per second".[1] While this is simply a metaphor, it is certainly true that features of a person's movement can communicate a portion of the message. The phrase "yay high", for example, must be accompanied by the appropriate hand position to indicate just how high (or low) the speaker means. The dynamic aspects of body language augment the spoken dialog yielding a much richer information channel. Similarly, in the video and film media the motion of the camera is often used in a stylized fashion to convey a particular abstract concept to the viewer. A zoom-out may indicate an ending, while a zoom in may indicate a beginning.

These observations on communication are not new. What is new is the ability for computer software designers to utilize the power of these media in both existing and new computer applications. Digital signal-processing hardware has brought high-quality stereo audio, natural image, and motion video to the desk-top workstation. The challenge before us now is how to combine the power of these "natural" I/O media with the interactivity of the computer to yield more effective computer applications.

Not Just For Bank Balances, Anymore

Looking at the historical development and fundamental changes in computing, we find that the original uses for computer were numeric in nature. The computer kept track of a bank balance or solved a numerical integration. Hollerith strings stored character text, but were not the subject of computation. Input and output were typically boxes of punch cards and deep stacks of printouts, both containing rows and columns of numbers.

The next evolutionary step was to symbolic processing. While usually associated with Artificial Intelligence, we include as symbolic computation that which employs

1 30 frames (or images) per second is the standard television display rate in the U.S. Each frame actually consists of two different fields, each of which present alternating lines of an interlaced image, so at 60 fields per second we could say "60,000 words per second".

pointers. Thus, modern data structures and data bases are symbolic in nature, not just the symbolic programming languages (LISP, Prolog, etc.). The human-computer interface came to include names (symbols) and relationships. Powerful operations on the symbolic structures, such as search, inference, extrapolation, were the trade-offs for the extra space and time required to store and process the symbolic information.

This brings us to the present day, in which the new orientation is simply storage (recording) and retrieval (playback) of noncoded information. Although understood in some sense at a very low level, the natural images and sounds are simply translated between the analog and digital domains. There is no understanding of the information contained. Indeed, the objects of which we are speaking are so rich in content that humans rarely agree on the meaning of, for example, a picture of a wind-blown wheat field. The computer will happily "capture" that image and reproduce it when bidden.

It is in this context that we develop the notion of computer communication enhanced by this interactive multimedia technology. The point is that it is an "author" (nearly always a human, today) who defines the ordering and synchronization of this "playback" in order to convey a message. The "answer", formerly a number then followed by a list, is now a multi-sensory "experience". The richness of information content of the media allows "transmission" of the answer from the author to the user.[2]

An Example, Audio Annotation for Electronic Mail

Electronic mail is a ready example. Widely available from many sources, electronic mail is extremely effective for rapid communication between people, including both acquaintances and people who have never met. Arpanet mail and bulletin boards, for example, link people around the world in all manner of organizations (Universities, Industry, etc). A fundamental limitation of this communication[3] is the text-based nature of the information. While a usable least-common-denominator, the constraints have lead people to invent extensions mechanisms for augmenting the text. Out-of-band information, e.g. the instructions "----- cut here -----" within a message for separating the attachment from the base message is one example. A more apropos example is the invention of the character-based icons for conveying emotional tone. Character-grams such as :-) and :-(convey happiness (or humor) and sadness, respectively.[4] Even with these annotations, it has been the author's experience, as others have found, that written messages are all too easy to misinterpret. So much of the usual information which people depend on in conversation is missing (pitch, timbre, inflection, timing). It is a wonder that the message gets through at all.

Contrast the problems of text-based mail with the power of multimedia mail. One simple example of the possibilities, utilizing text, graphics, animation, and audio is the commercially available product Freestyle from Wang. Functionally, this product allows the user to capture a textual screen from any running application and add hand-drawn text and graphics along with audio annotation (usually speech). The recipient of the electronic mail message may then play back the annotation in "real-time". One hears the author's voice synchronized with the "ghost writing" exactly as they were recorded. Similar audio-annotated communication is available in the Macintosh and NeXT environments.

It did not take the author long to be convinced of the power and increased effectiveness of communication in this fashion. All the elements of normal telephone conversation are present,[5] plus the added dimension of the real-time handwriting playback. The visual clues, the body language, are all that are missing, and digital video compression hardware will pave the way for that, too.

Hypermedia Design

Given the potential capability of this communication medium, the question remains: "How do we use it effectively?". With a set of basic output elements (still and moving images, stored audio, text) and input devices (mice, joysticks, keyboards, touch screens, cameras, microphones) the number of possible combinations presents many opportunities for poor design. As Desktop Publishing made possible documents which use every font on every page, rendering it unreadable, there is potential for many unwatchable and unusable interactive hypermedia applications. We are reminded of the typical "home movie" - jerky images and awkward action

2 We feel that all of the current terms, "user", "viewer", etc., are inadequate for describing the person at the receiving end of the interactive multimedia experience. We have settled on "user" as a poor compromise.

3 Although it is hard to describe such a powerful medium as "limited".

4 To see this, tilt your head to the left to see the eyes : , nose - , and mouth) or (.

5 Except the interactivity, making this more like answering-machine-mediated communication.

("Everybody, wave at the camera!"). With the availability of consumer-grade video editing equipment, however, we find increasing sophistication in the story telling ability of home video. People are learning, with broadcast TV as a model, how to design in this new medium in order to "get the message across".

While this last example speaks to creating home videos, we feel that there are already-established schools of design for many media: video, film, graphics, drama. The introductory course for any of these domains, e.g. Graphic Design 101, teaches the basic principles for creating good designs.[6] With the addition of the interactive capability of the computer for hyper-link branching, we feel that a new set of design rules will arise from a synthesis of the rules from the various media which make it up. The result will be a course entitled "Hypermedia 101", and will address the issues arising from combination and coordination of the various output and input channels available, as well as issues of branching, including the underlying logical structure of the information being presented.

A key aspect to be treated in these guidelines is the development of involvement in the user - how to "draw" the user into the application. Such techniques already exist in various forms for the current media, for example "characterization" from film, stage, and literature. The creation of a persona with whom the user can identify and grows to care about is a powerful way to bring about involvement. Exploiting the "conversation" (interaction) as part of the user interface will be the challenge of design in this new medium.

Hyperchannel Communication

If an anthropologist from Mars were to land on Earth what theories could they derive from studying a current workstation or personal computer?[7] Imagine that all pictures of what we humans look like were mysteriously destroyed. What sort of creature would be re-constructed? It would have a very weak spine, requiring it to sit, constantly; monocular vision with very limited sensitivity to color; three hands (two for the keyboard, noting the symmetry of the left and right shift keys, plus one hand for the mouse) with very limited range of motion; and very poor hearing. We should ask why this reflection of ourselves in our technology is so far off base!

In this regard, perhaps hypermedia systems should instead be termed "hypersensory", as this is in fact one basis for the power and potential effectiveness of the technology. The justification for the increased cost of hypermedia in terms of storage and processing power is the more appropriate match-up in capabilities between the computer and the human. Observing what we, today, think of as a modern hypermedia application (with rich graphics, moving pictures, high-quality sound), our alien anthropologist would get a much more accurate picture of our sensory capabilities. If the application were also of the "virtual reality" genre utilizing stereoscopic "eye phones", stereo ear phones, a data glove, speech recognition/generation then the human portrayal would be much more accurate. This is not surprising, as virtual reality researchers[8] have often spoken of the explicit design goal of fully utilizing the human sensory capability in the user interface.

Experimenters at Xerox EuroPARC have been exploring audio output as part of an "Alternate Reality" environment. In the ARKola[9] experiment test subjects work jointly at different computers to run a simulated beverage-bottling plant. The graphical representation of the plant is manipulated so that the entire plant would not fit on the screen at one time. In the natural model which evolved for collaboration between the two subjects each focused their view on one-half of the plant and communicated between themselves to establish coordinated actions. In one test group of subject pairings, sound effects provided feedback on the state of operation of the plant. In the other test group, the application was silent. The finding was that using the sound of the operation of the non-visible portion of the plant did improve problem solving ability.

Each user receives cues on the actions (and their effects) of the partner which were directly coordinated with the operation of their portion of the plant. The extra audio information enhanced each user's internal model of the domain and improved the problem solving ability of the team. This example has obvious implications for cooperative work environments such as showing how the limitations of screen real-estate of the visual medium may be attacked using a multimedia approach.

As well as using the different I/O channels of a hypermedia system in parallel to convey different aspects of a single message, we may also use the various media for several different messages simultaneously. Without hypermedia, notification of asynchronous events, such as the arrival of new electronic mail, may be announced by popping up a window on the display

[6] Of course sophisticated designers will break these rules and still be effective, but this is based on training, experience, and talent.
[7] Thanks to Bill Buxton for this allegory.
[8] Such as Jarad Lanier of VPL.
[9] Demonstrated by Bill Buxton and Bill Gaver as part of the tutorial "Non-Speech Audio", Chi '90, Seattle, Washington.

screen. Given audio output, especially speech generation, the visual environment may be left undisturbed, using the aural channels to convey the notification. The choice of a particular channel gives the application designer new freedom to tailor the user interface to the semantic content of the data. While human capability for processing multiple distinct is limited, and it remains to be seen how fully this parallel communication may be exploited, simply removing the steps (keystrokes, mouse clicks, etc.) to close the notification window will be an improvement. Another very simple and compelling example[10] is the ability for the computer to choose voice communication over visual for the situation when the user is across the office not attending to the computer screen at all.

Another Example, John Cocke: A Retrospective by Friends

In our recent work, we[11] have been focusing on the effective combination of motion video (TV) with interactive branching. We undertook the John Cocke project as way of learning-by-doing what some of the parameters and boundaries are in design for this area of hypermedia. We are incorporating our discoveries pertaining to both application and tool design in our ongoing research.

Commissioned for a symposium honoring IBM Fellow John Cocke's 35th anniversary with IBM, we developed an interactive laser disc application depicting the man and his career. He has had a very rich history with IBM, has worked on many key advanced projects, and is recognized as originating many of the fundamental ideas behind compiler optimization, high performance computer design, the RISC architecture concept, among many others. He also has a unique personality and is warmly regarded (loved, actually) by all those that come to know him.

From the beginning, we felt that the interactivity of the computer combined with the power of video would help us with the difficult task of capturing the diversity and complexity of John Cocke. The history was told through video-taped interviews of 14 colleagues of John's, as well as John himself. Much in the style of a film or video documentary, we extracted the "choice" portions of the 22 hours of interview footage and boiled it down to a 1 hour laser disc.[12] Unlike a video documentary, however, the selected bits (termed "sound bites") were not woven into a single, linear piece. Instead, a hypermedia structure was designed to organize the video from several different perspectives:

Who by each interview, grouped by topics "The Man", "His Work", "Impact", "Style", "Stories".
What by major system project
Where places from John's history
When a time line by year
How how John does what he does, his personal characteristics
Why his importance to IBM, including key contributions, significant awards (e.g. the ACM Turing award)

Another top-level view was essentially a random organization. Called **Quiz**, this set of 35 trivia questions about John Cocke served both to give a general feel for the data (video clips) and also appealed to the entertainment aspect of the banquet and symposium. The multiple-choice questions in the quiz were formed by selecting interesting answers from the available material and then choosing the question to fit. An incorrect answer yielded a video of one of the subjects on the laser disc saying, "I'm afraid that's incorrect", or "You must mean X" where X is the correct answer. These positive, negative, and hint feedback pieces were taped long before the questions were designed. As with the answers, the available feedback shaped the design of the questions to some degree. Finally, a small section gave further explanation of the different perspectives (**Help**, for when our interactive design failed to be intuitively obvious) and described the underlying technology.

Presented using a touch-screen, laser disc, and video windowing adapter,[13] the user interface was styled as a tree of multiple choice menus with graphics and video stills combined. The leaf nodes of the tree consisted of video "sound bites". Early user testing suggested that it was important not to build the tree too deep, requiring a long sequence of menu choices to get to the "reward" video segment. We flattened the tree accordingly, taking at most three choices to reach a leaf. Testing also pointed out the motivational states for different users. Some users took more active control, navigating easily through the menus. Others preferred the information be presented to them by the system, with the user taking a more passive role.[14] This prompted us to expand the "attract mode" (as in video games) portion of the application, intended to play

10 Described by Nicholas Negreponti.
11 The Interactive Media Project, IBM Thomas J. Watson Research Center.
12 This selection and editing process is the subject of a set of papers currently in preparation.
13 IBM's M-Motion Video Adapter.
14 Users cited unfamiliarity with the information, and therefore no good basis for making navigation choices, as one reason for taking the passive role.

sequentially through the material when no one is using the kiosk. Originally a brief selection, the attract mode grew to cover much more of the material taking approximately 1/4 hour without repetition. The user may take control at any time simply by pressing the touch screen.

The result is an attractive, interactive presentation which gives a well-rounded view of the man, John Cocke. The user receives a view of his career with IBM, his impact on not just IBM but the entire field of Computer Science, and also very warm, personal accounts of his unique style. The video interviews with his colleagues and friends gave a richness in variation and historical feel to the account, a "Retrospective". The number of people we interviewed, many of whom used surprisingly similar terms to describe John, helped give weight to the individual comments. And, finally, the use of video clips gives the feel that the people on the laser disc are speaking directly to the user.

Conclusions

We in hypermedia research are exploring the use of the recently available digital hardware which brings rich analog media to the desktop. The challenge lies in designing for this new communication medium which borrows from film, TV, literature, and adds interactivity or branching. The potential power of the richness and realism in the user interface will provide an enhanced communication channel between designer and user; user and peer user; computational model and user. The input ability of the channel will allow consideration of the user's state of mind, such as a history of the recent screen touches, a joystick input device for continuous indication of the user's interest level, or visual sensing of the user's body language. We will learn, over time, how to design systems which better match the sensory capabilities, vision, motion, sound, touch[15] of the people who use them.

[15] Some day perhaps taste and smell, too.

N92-21954

From Knowledge Presentation to Knowledge Representation to Knowledge Construction: Future Directions for Hypermedia

by

David B. Palumbo
The University of Houston-Clear Lake

presented at the Hypermedia '90 Conference
Houston, Texas
December 5, 1990

A K T
ADVANCED
KNOWLEDGE
TRANSFER
G R O U P

Houston, TX • 77058-1056
(713) 283-3565

Introduction

Much of the excitement surrounding hypermedia systems is their ability to meet the needs of various users. These include authors, designers, on-line readers and others using the idea processing capabilities of such systems (Marshall, 1987). The central theme of currently available systems is knowledge presentation. However to fulfill their promise hypermedia systems need to move toward more sophisticated interpretations of knowledge representation and finally toward knowledge construction. This paper begins with a discussion of the relationships between human memory systems and hypermedia system with particular emphasis on the underlying importance of associational memory. Then the distinctions between knowledge presentation, knowledge representation, and knowledge construction are addressed. Finally issues involved in actually developing individualizable hypermedia based knowledge construction tools are presented.

Parallels Between Human Memory and Hypermedia

Much is made of the similarities of hypermedia-based systems and current conceptions of human memory. These human memory models are primarily based on information processing theory. In this section we will examine this relationship as well as discuss strengths and weaknesses inherent in these current analogies between hypermedia and human memory.

Similarities

Current conceptions of learning are based on principles of cognitive psychology. Learning can be defined as the reorganization of knowledge in semantic memory (Jonassen, 1988) The interconnections of knowledge in a structured associative network allow learners to combined ideas, extrapolate, and infer. These structural networks are composed of both the information presented as well as the relational links which interconnect them (Norman, Gentner, & Stevens, 1976). Based on this description of semantic networks, learning can more explicitly be described as building new knowledge nodes and connecting them with existing ones and with each other (Norman, 1976). The stronger the connection between the existing knowledge stored in memory and the newly acquired knowledge, the better the information will be learned. Learning, therefore, becomes a function of connecting new material onto one's preexisting knowledge structure (Jonassen, 1988).

If we accept this cognitive definition of learning as a reorganization of cognitive structure, then, we need access to tools for assessing cognitive structure, tools for depicting and displaying appropriate knowledge structures, and ways of mapping that structure onto the learner's existing knowledge structure. Current computer environments, especially those based on hypermedia, are capable of doing this. In fact much of the excitement surrounding hypermedia's potential centers on its use as such a tool.

From this description, it is clear to see the common terminology used by both cognitive psychology in describing the operation of human memory and hypermedia systems. Nodes and links form the basic structure of each. In fact the human memory model is currently based strongly on a computer analogy, comparing the storage and retrieval from human memory with similar mechanisms in computer-based technologies. Hypermedia extends this notion by allowing for a more explicit relationship between information in a computerized information base. Associations between information, a key aspect of human memory, are also central to hypermedia.

Rumelhart (1977) points out that the essential attribute of the human memory system is not the storage or retrieval of specific units of knowledge, but rather the organizational schemes by which knowledge is associatively related. Hypermedia has provided a computerized technology to achieve similar relationships. A second fundamental aspect of human memory is that when new associations and therefore new organizational schemes are developed, it is not necessary to completely recodify the prior knowledge within the newly acquired structure. Hypermedia environments also provide such flexibility in computer information systems, in that new information and new relationships can be easily integrated into previously stored information without having to recode that information.

Human memory also utilizes a variety of organizational schemes, not just on general scheme, to store and retrieve the variety of knowledge presented. Research has demonstrated that the human memory system stores and structures information and associational schemes that preserve the most important aspects of the associations, yet does not preserve other possible association (Bransford & Franks, 1971; Bransford, Barclay, & Franks, 1972). Hypermedia systems offer the possibility of similar organizational schemes, allowing the the designer and/or the user to decide on relevant relationships between information, while not attending to other possible, yet less important relationships.

Differences

It is worthwhile to note that prior to the inception of the computer as an acceptable metaphor for human memory models, the library was used as the prime analogy. However, both the computer, especially when used as a hypermedia device, and the library metaphor breakdown at certain points in their respective parallels with the human brain. For instance, Rumelhart (1977) points out that while other information storage systems, such as a library of books stores "complete" units of information, the human brain appears to store fragmented bits of information which, must then be somehow processed via the retrieval system to form a complete unit of knowledge and therefore allow the answer to a specific query of the human knowledge base.

Hypermedia systems, like a library of books, store complete information "chunks" in each node. If fact, it is now a point of contention whether such systems should have the capability to contain more than one complete "chunk" of information per node. A second issue is whether, in hypermedia systems, the nodes should be seen as repositories of these units of informations or should the nodes be constructed as the information itself.

Thus the retrieval of information from the human memory system can be broken down into two equally important processes: first, the location of the desired information; and secondly, the reconstruction of appropriate output from the incomplete information stored. Hypermedia systems differ in that they do not accommodate this second aspect since they traditionally hold complete units of information, and therefore no processing of information is required to provide an answer to a specific query.

Linking Information

Information in both human memory and hypermedia is related through associational links. Much of the concern in developing current hypermedia systems focuses on the underlying problem of using merely associative links. Most systems support only that one unit of information is somehow related to another unit of information. Human memory supports a much stronger linking mechanism, in that the links also convey information about the associational relationship.

This is especially evident in the placement of nodes of information in a large hyperspace. An underlying assumption of hypermedia designers, which is often then passed on to users is that distance between nodes in hyperspace is directly related to the strength of their association (Locatis, Letourneau, & Banvard, 1989). Yet this is not necessarily the case. And reliance on such a metaphor may increase both the development cycle, and therefore the cost, of a hypermedia, as well as the cognitive load associated with using the system. Since there is still no consensus in the brain physiology of information storage and retrieval, no such reliance on distance is present in human memory. The strength of the relationships are conveyed by the value of the associational relationships.

Hypermedia systems that allow for typed link relationships may alleviate much of this problem in that a designer can connote the strength of a relationship by the type of link used to connect two nodes of information, regardless of the distance between them in hyperspace.

Conclusion

While information processing models of human memory and hypermedia share many common features that seem relevant in assessing the potential impact of hypermedia in learning environments, it is clear that there are certain differences that prevent one from asserting that hypermedia is simply a computerized information processing system that parallels its human equivalent. The importance of associational memories in both systems merits a closer look at common features shared by all associative memories. These include that:

(1) they can recall information based on incomplete or garbled inputs

(2) they can store information in a distributed fashion

(3) they display some degree of content addressability

(4) they are strongly robust in that they do not degrade appreciably when nodes and/or links are lost or information is input inaccurately

(5) they can generalize between information, both in terms of content and structure, and previously stored information

(Caudill & Butler, 1990)

While hypermedia proponents have based much of their theoretical claims on parallels between hypermedia and associative memory, one can quickly see that current hypermedia systems do not meet all of these central requirements. If the relationship between hypermedia and associative memories is critical, then the next generation of these system should focus on meeting these essential characteristics.

Knowledge Presentation, Knowledge Representation, and Knowledge Construction

Much of the discussion about the impact potential of hypermedia has centered on the ways that such systems may become infused into our society. Current systems tend to focus on either the presentation of information or the representation of information in an advanced storage and retrieval system. Others propose a next generation of hypermedia that will focus on the the construction of knowledge.

The power of hypermedia applications is seen in the following three characteristics that relate directly to their use as presentation tools, representation tools, and construction tools (Collier, 1987):

(1) Printed knowledge is inherently nonlinear and often has arbitrary ordering forced on it by the print medium. Hypermedia systems eliminate such constrains in the presentation of information. Such benefits relate directly to hypermedia as a knowledge presentation environment.

(2) Semantically and logically related information can be tied together in conceptual webs. This benefit draws heavily from the parallel between hypermedia systems and human memory and is explicitly related to the power of hypermedia to structure and represent knowledge in an associational network similar to the function of the human brain.

(3) Other means for making connections among information support only part of the potential web of interconnections. Since their is no way to fully anticipate the prior knowledge, experiences, and learning style of a potential user, other information systems are limited in that users may be unable to adequately transfer desired information into their existing cognitive structure. Hypermedia, on the other hand, holds the potential to allow users access to the tools by which they can construction the transitions between the information to be accessed and their cognitive structure; thus, truly individualizing the learning environment.

Hypermedia as a Presentation System

As a presentation system, the ability of hypermedia systems to show or exhibit information in a multimedia framework is emphasized. In fact much of the excitement of lower end hypermedia systems, such as Hypercard and Supercard, tends to focus on their multimedia aspects rather than the non-linear attributes critical to any hypermedia system.

The emphasis of hypermedia as a presentation system is exemplified in Oren's (1987) discussion involving the notion that the designers of hypermedia systems should focus on construction of the most useful pathway for the user to proceed through the information in a particular hypermedia. Thus, he notes that hypermedia design should anticipate the needs of the learner and present information accordingly.

However, one must note that simply because hypermedia systems appear to be good vehicles for capturing, structuring, and presenting information, such attributes do not necessitate that these will be used to their fullest potential in the development of hypermedia-based knowledge representation systems.

Proponents have addresses several ways in which the Notecard hypermedia system can be extended to more fully move from a multimedia presentation system to a more sophisticated knowledge representation system. One is that such systems will need to become more adept at formalizing the representational process within the system.

Hypermedia as Knowledge Representation

As a representation system, much is made of the similarity of hypermedia to current models of long term memory as previously discussed. In fact, the definition of representation as the capacity to picture to the mind a mental image or idea, leads one to such parallels. Certainly there is a common terminology that also promotes such a relationship. Nodes and links are the metaphor for both. Nodes and links are also the common ground of artificial intelligence and linguistics researchers. Yet researchers in these disciplines have been hesitant to claim that they are referring to the same entities. In fact the field of cognitive science has evolved to reconcile the psychological, linguistic, and computer conceptions of knowledge representation and promote a more multidisciplinary approach to study in this important area.

In fact, researchers are beginning to see that while one of the often tauted aspects of hypermedia is their ability to support the emergent properties of the representation process, current hypermedia systems have failed to develop these opportunities. Specific inquiry into the fundamental aspects of nodes and links are needed if hypermedia is to become a sophisticated knowledge representation system.

Current systems differ in the way information is related to the nodes of a hypermedia. One difference is that in some systems, such as the IRIS Intermedia program, the information is stored as nodes. Other systems, such as the Thoth-II systems, separate the nodes and the information they contain. The benefit of this second type of system is that they allow for more than simple connection between units of information by allowing the conception of the knowledge representation to be conveyed from the designer to the user (Collier, 1987).

Hypermedia systems also differ in the amount of information that may be placed in the nodes of this second type of system. One type, exemplified by Textnet, allows only one unit of information to be placed in a particular node. The principle behind the Thoth-II system, on the other hand, is to allow for multiple units of knowledge to be placed in any node.

The use of linkages in hypermedia is also a critical issue as such systems move from mere presenters of information, to more sophisticated knowledge representational systems. "In many representations, a key decision centers around the distribution of meaning- should links or cards carry the semantic burden" (Mitchell, 1987, p.265). The semantic weight of a hypermedia needs to be equably distributed between its nodes and links as neither entity is capable of supporting the full semantic load alone.

While initially much of this weight was placed on the nodes of the network, current implementations are moving more of this burden to network links. The possibility of making value a link property would be beneficial in developing more complete knowledge representation systems in hypermedia. However, performing a representational task or interpreting the results of an analysis may become confusing if link types are used for too many semantically orthogonal purposes (Mitchell, 1987).

One future direction of hypermedia is to develop systems that are capable of capturing knowledge representations via some type of concrete structure that could then be reapplied to other knowledge bases (Mitchell, 1987).

Hypermedia as Knowledge Construction

Another key claim of hypermedia proponents is that these systems will be effective as a teaching medium by allowing users to individually access a large knowledge base and seek out relevant information that meets their particular needs, both in terms of their prior knowledge as well as their preferred learning style. The development of systems to achieve these ends is still a possibility. However, there is little empirical evidence that by simply providing an advanced presentation system, or even a more elaborate information storage and retrieval system that parallels the way that the human brain seems to represent knowledge, that more effective or efficient learning will occur (Locatis, Letourneau, & Banvard, 1989). A more constructivist environment, where the user not only browses the information base, but also has the ability to build additional nodes and linkages, holds more promise to promote learning. Many hypermedia systems support such an environment, yet little has been does to promote this obvious advantage.

Raskin (1987) laments that hypermedia has been heralded with mostly uncritical attention. And while he does state that current implementations of hypermedia are worth pursuing, he strongly cautions that they may fail to realize the expectations currently promised. His criticisms, however, focus mainly on technological and user-interface design limitations that seem addressable in the near term. However this rationale can also serve as the basis of more daunting concerns in that current directions in hypermedia development focus on the presentation aspects and storage/retrieval capabilities inherent in such systems, while to make a more substantial impact hypermedia systems need to focus on allowing users to actively construct information, via typed linkages. The potential of such systems is more strongly grounded in psychological literature on learning and transfer.

A key issue in the emergence of hypermedia is the ability of these systems to promote learning in an effective and efficient manner. In fact, the term HAI (hypermedia assisted instruction) has been proposed to describe the use of such systems (Heller, 1990). While it is beneficial to extend beyond the traditional uses of computers in instructional settings (e.g. drill and practice and tutorial remediation) inherent in Heller's rationale is that current hypermedia systems are incomplete and need to be augmented to meet this challenge. The issues that she addresses also focus on presentation and user interface issues. A more important issue that hypermedia developers need to address, especially within the cognitive paradigm proposed by Heller, is in allowing the user to construct knowledge from within the hypermedia environment.

Individualized Learning Environments

The ability to individualize information access to accommodate the diversity of possible users has been traditionally seem as a stronghold of computerized environments. As our society continues to evolve into a more global one where accommodating only the ethic and cultural majority no longer proves effective, technologies that transparently accommodate the differences inherent in this global society are needed.

Computer assisted instructional environments first offered the ability to individualize information access. Such systems, however, are limited in that they can realistically only accommodate differences in the rate at which a variety of users progress through the information base. More sophisticated individualized systems are necessary, and proponents of hypermedia hold hope that hypermedia based systems will provide the environment to truly accommodate the evolving needs of a global information society.

Hypermedia and Learning Styles

Research has supported the claim that cultural influences have an effect on the cognitive learning styles exhibited by individuals (Ramirez & Price-Williams, 1974; Witkin, 1967). Learners' cultural background may effect differences in both their intellectual skills and intellectual performance. Children of different cultural and linguistic groups exhibit significant variations in both the cognitive and sensory perceptions.

Cohen (1969) has identified two basic learning styles, analytic and relational. Those who learn in an analytic style view information as part specific, objective, and tend to view information as it is, rather than in some context. Those who exhibit a relational learning style focus on a more global context and in a subjective form. They also tend to view information in its own context. Kirby (1979) points that to address the cognitive learning styles of all learners, information environments should be structured bicognitively since users who do not function effectively in the currently practiced analytically structured environment will be poor achievers and also will become successively worse.

A crucial, and yet often neglected, aspect of effective information transfer is ascertaining users' learning styles and then accommodating them accordingly (Ausubel, 1968). Research suggests that learners who were taught by their preferred method achieved better, were more interested in the subject matter, liked the way the subject was taught, and wanted to interact with other subjects in the same way (Smith & Rezulli, 1984). Matching presentation style of the information with the desired learning style of the user enhances cognitive outcomes. Therefore, by taking users' learning styles into consideration, they may become more involved in the learning process.

Hypermedia as an Instructional Environment

Hypermedia based systems allow the redefinition of both the structure and content of the material to be learned. This ability alters the constraints and opportunities for conveying information when compared to traditional forms of information presentation. The power of such a tool can be seen as both subtle and incremental; yet we need to harness this power to effectively and efficiently develop training programs that meet the requirements of the information age (Scacchi, 1988).

In traditional forms of instruction, learners most often are presented with information in a sequentially formatted environment. Hypermedia, on the other hand, allows the learner to access any information in the knowledge base (Jonassen, 1988). Learners need not be constrained by the structure imposed by either the information or the instructor. Since each learner has an unique knowledge structure based on their experiences and abilities, the way that they choose to access, interact, and interrelate information in the knowledge base will also vary. Hypermedia based learning environments allow the knowledge base to accommodate the learner rather than the learner accommodating the knowledge base.

In allowing for maximum use of this type of environment, the learners should be encouraged to explore information, make associated links and relationships and even alter the knowledge base to make more sense from their previous experiences and learning style. Hypermedia offers the potential to construct an environment that allows for these beneficial activities (Jonassen, 1986).

A major characteristic of hypermedia environments is that they allow users to link information together in many ways and to make these relationships obvious as well as the conceptual relationships that they describe. Instructors and learners may create different pathways through the hypermedia knowledge base. Users can also annotate the knowledge base by creating notes, explanations, and analogies. A major goal of hypermedia is to provide a learning environment that facilitates exploration (Jonassen, 1988). This type of learning environment provides immediate access to large collections of information. The most distinct aspect of hypermedia learning environments is their ability in a node-link framework based upon semantic structures, to portray an accurate structural description of the knowledge base they are representing.

Hypermedia offers advances from previously available technologies in that it is strongly connected with a cognitive conceptual framework, yet this framework does not limit or constrain it possible application (Jonassen, 1988).

Cognitive Load

A second benefit of hypermedia-based individualized learning environments is in the possibility of decreasing the cognitive load associated with accessing information from within such an environment. Any information presentation/retrieval system has some load associated with its operation. Users must accommodate issues of learnibility, efficiency, ease of remembering, and error frequency. The amount of time a user must devote to such system operational issues, directly increases the amount of time and cognitive energy required to effectively interact with the information system. Efficiency of use is also adversely affected. Therefore, systems that decrease the cognitive load induced by the system will allow for more efficient use of the system.

Nielsen (1990) addresses five usability parameters that are directly related to cognitive load. These include the ease to which the operation of the hypermedia system is learned; how efficiently the system can be used once the user has learned its effective operational structure; how easily the operation of the system is remembered from one interaction to the next; the number and cost of errors associated with system operation; and how pleasant the system is to use.

Certainly, if hypermedia systems can more effectively accommodate the usability parameters addressed above, then they would also decrease the cognitive load when compared with other methods of information access. However, while some are praising hypermedia in this area, others point to cognitive load as one of the largest drawbacks of hypermedia environments.

The question of how much and at what level information should be presented to the user is often at the heart of such concerns. Issues of how many simultaneously displayed nodes should be allowed on any given screen and the how many links should any one node of information support are questions that need further investigation. To see how strongly this issue is tied to the issue of cognitive load, one of the prevailing sentiments in this area is that the number of nodes displayed and the the number of links allowed per node should be limited to seven; a direct connection to Miller's (1956) assessment of the limits of human working memory.

Novice/Expert Users of Hypermedia

Another key issue in the use of hypermedia is the prior expertise and knowledge requirements of the intended user. While knowledge presentation systems may be very useful to those considered expert in the content area of a particular hypermedia, such presentation systems do not hold the key learning tools required by those non-experts when relating to a particular knowledge base.

While there is a clear continuum between novices and experts in a particular knowledge area, a distinction between experts and non-experts is appropriate in interpreting the potential of hypermedia as a learning tool. Issues of cognitive overload, user disorientation, superficial browsing and disinterest often reported by users of hypermedia may well center on the issue of the level of experience of the user. Thus while current hypermedia systems may well decrease the cognitive load of those users closer to the expert end of the continuum, they may well increase the load on more novice users.

Summary: Two Directions for Hypermedia

This paper has addressed two possible future directions for hypermedia, both of which hold promise, yet need further investigation if hypermedia is to become more than just another "hyped" media (Locatis, Letourneau, Banvard, 1989).

The Next Generation Database?

While the focus of this paper centers on the movement from the storage and retrieval capabilities of hypermedia to a more constructionist learning environment, hypermedia does possess attributes that may lead to a next generation of database, one whose major characteristics include hypermedia. Such systems would clearly be useful to any number of users. As we move head long into the information age, an important attribute of the work force will be the ability to access information; as it will no longer be important what information one possesses, only how efficiently they can access the desired information.

Current work in hypermedia seems to focus on this direction and much of the current criticism of available systems rests on the inability of hypermedia users to access such large volumes of knowledge in efficient ways. Organizational aspects of hypermedia are now a central development issue (Conklin, 1987, Halasz, 1988). Some system designers have moved toward a hierarchical linking structure, where movement between information nodes at one level of the hypermedia is restricted to access only those nodes directly above or below it in the designers structure. Other systems support referential linking, where any two nodes can be linked together. Certainly this second type of systems, while more difficult to construct, especially if the designer is to construct all meaningful linkages, meets more the central attributes of non-linearity in hypermedia development and would be critical if a new generation of databases of information centered on hypermedia are to become a reality.

Another issue that makes the possibility of this direction seem more reasonable is the requirement for the next generation databases to contain more that textual information. Hypermedia, with its multimedia capabilities seems ideal to allow database-like retrieval of textual, graphic, auditory, and filmic information.

Current proponents of this type of hypermedia development also stress other benefits that hypermedia offers in this area. These include the ability to mix both highly structures and loosely structured information together. They also would allow for multiple representations of the same information. And they would allow for the extension of the information base in ways that may not conform to the original pattern (Marshall, 1987).

Knowledge Construction Sets

Initial hypermedia systems such as Notecards, IBIS, and Intermedia required large computer systems. The recent introduction of microcomputer-based hypermedia systems such as Hypercard, Linkways, and Guide have substantively contributed to the hype surrounding hypermedia. However, these microcomputer-based systems have focused more on the presentation of materials rather than the instructional applications that hypermedia may promote. Systems such as Hypercard are often referred to as programming constructor sets, where a user with little computer programming experience can successfully produce a functional piece of software with minimal effort do to the ease to which their scripting environments can be mastered.

The promise of hypermedia, however, does not revolve around an easy way to produce software. Instead, rather than working to promote programming constructor sets, proponents of hypermedia need to focus on developing knowledge constructor sets. Environments where information presentations can successfully and efficiently be transferred into knowledge to a diverse and every changing population of learners.

Much of the theoretical framework for hypermedia promotes the development of such systems. Yet, little has been done to support such implementations. Future work in the area of hypermedia needs to address the movement of hypermedia systems into the area of cognitive science, issues of transfer of training from the hypermedia to the learner, and the incorporation of artificial intelligent systems within hypermedia information bases that will effectively and efficiently allow learners of all experience levels, abilities, and learning styles to the interact with the information environment.

References

Ausubel, D. (1968). *Educational psychology: A cognitive view.* New York: Holt, Rinehart and Winston.

Bransford, J. D., & Franks, J. J. (1971). The abstraction of linguistic ideas. *Cognitive Psychology, 2,* 331-350.

Bransford, J. D., Barclay, J. R., & Franks, J. J. (1972). Sentence memory: A constructive versus interpretive approach. *Cognitive Psychology, 3,* 193-209.

Caudill, M., & Butler, C. (1990). *Naturally intelligent systems.* Cambridge, Massachusetts: MIT Press.

Cohen, R. (1969). Conceptual styles, culture conflict, and nonverbal tests of intelligence. *American Anthropologist, 71,* 828-856.

Collier, G. H. (1987). Thoth-II: Hypertext with explicit semantics. *Hypertext '87 Proceedings* (pp. 269-290), ACM Press.

Conklin, J. (1987). Hypertext: An introduction and survey. *IEEE Computer, 20*(9). 17-41.

Halasz, F. G. (1987). Reflections on Notecards: Seven issues for the next generation of hypermedia systems. *Hypertext '87 Proceedings* (pp. 345-366), ACM Press.

Jonassen, D. H. (1986). Hypertext principles for text and courseware design. *Educational Psychologist, 21* (4), 269-292.

Jonassen, D. H. (1988, November). Designing structured hypertext and structuring access to hypertext. *Educational Technology,* 13-16.

Kirby, P. (1979). *Cognitive style, learning style, and transfer skill acquisition.* Information Series Number 195. Columbus: Ohio State University, National Center for Research in Vocational Education.

Kirby, P. (1979). *Cognitive style, learning style, and transfer skill acquisition.* Information Series Number 195. Columbus: Ohio State University, National Center for Research in Vocational Education.

Locatis, C., Letourneau, G., & Banvard, R. (1989). Hypermedia and instruction. *Educational Technology Research and Development, 37* (4), 65-77.

Mitchell (1987). Exploring representation problems using hypermedia. *Hypertext '87 Proceedings* (pp. 253-268), ACM Press.

Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychology Review, 63*, 81-97.

Nielsen, J. (1990). *Hypertext and Hypermedia*. Boston: Academic Press.

Norman, D. A. (1976). *Studies in learning and self-contained education systems*, 1973-1976. Technical Report Number 7601. Washington, D. C.: Office of Naval Research, Advanced Research Project Agency. (ED 121 786)

Norman, D. A., Gentner, S. and Stevens, A. L. (1976). Comments on learning schemata and memory representation. In D. Klahr (Ed.). *Cognition and instruction*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Oren, T. (1987). The architecture of static hypertext. *Hypertext '87 Proceedings* (pp. 291-306), ACM Press.

Ramirez, M., & Price-Williams, D. (1974). Cognitive styles in children: Two Mexican communities. *Interamerican Journal of Psychology, 8*, 93-101.

Raskin, J. (1987). The hype in hypertext: A critique. *Hypertext '87 Proceedings* (pp. 325-330), ACM Press.

Rumelhart, D. E. (1977). *Introduction to human information processing*. New York: John Wiley & Sons.

Scacchi, W. (1988). On the power of domain-specific hypertext environments. *Journal of the American Society for Information Science*.

Smith & Rezulli (1984). Learning style preferences: A practical approach for teachers. *Theory Into Practice, 23*, 44-50.

Witkin, H. A. (1967). A cognitive style approach to cross-cultural research. *International Journal of Psychology, 2*, 223-250.

SEPEC
Software Engineering Professional Education Center
University of Houston-Clear Lake
2700 Bay Area Blvd., Box 258
Houston, Texas 77058
(713) 282-2223